Day Ahead Load Forecasting for the Modern Distribution Network - A Tasmanian Case Study

Michael Jurasovic
School of Engineering
University of Tasmania
Hobart, Australia
mjj4@utas.edu.au

Evan Franklin
School of Engineering
University of Tasmania
Hobart, Australia
evan.franklin@utas.edu.au

Michael Negnevitsky
School of Engineering
University of Tasmania
Hobart, Australia
michael.negnevitsky@utas.edu.au

Paul Scott
Research School of Computer Science
Australian National University
Canberra, Australia
paul.scott@anu.edu.au

***Abstract*—Penetration of distributed energy resources in distribution networks is predicted to increase dramatically in the next seven years, bringing with it the opportunity for utilities to have a greater presence at low levels of the network. To achieve this effectively, utilities will require accurate short term load forecasts. This paper presents a novel neural network-based load forecasting system that applies recent advances in neural attention mechanisms. The forecasting system is trained and assessed on ten years of historical half-hourly load, weather, and calendar data to produce a 24-hour horizon half-hourly online forecast. When forecasting during anomalous peak holiday periods on a feeder that has a typical load of less than 1000kVA the forecasting system achieves a MAPE of 7.4% and a mean error of -15kVA. The forecasting system is implemented in a residential battery trial and is able to successfully forecast major peaks with sufficient lead time and accuracy to enable the fleet of batteries to charge ahead of time and provide network support.**

***Index Terms*—load forecasting, machine learning, DER**

I. Introduction

Electricity distribution networks, and the way in which they are managed, are currently going through a significant transition, with perhaps more change over the last ten years than in the previous hundred. Until recently, generation and load were largely viewed and managed separately: power was produced almost exclusively by large, centralised generating units, and was consumed by customers after routing via the transmission and distribution networks. Networks were designed and built for demand profiles which were relatively stable over time, with demand forecasting at distribution feeder level required primarily for long term network planning purposes only. Increasingly, power is both consumed by customers connected to the distribution network and is also generated and manipulated by distributed energy resources (DER) within the distribution network, often behind the meter of individual consumers. The impact of increasing levels of DER in the system creates, among other things, both a need and an opportunity to more actively manage distribution networks, while also resulting in a generally less predictable net demand profile.

DERs are controllable devices in the power network that generate, store, and/or consume power. This includes solar photovoltaic generation (PV), battery storage, and electrical vehicles. In Australia, the dominant DER technology deployed to date is solar PV, with over 1.8 million systems now reportedly installed on residential properties [1]. It is widely anticipated that battery storage and electrical vehicle uptake may be just as rapid [2].

The Tasmanian distribution network, meanwhile, is forecast to experience significant increases in these technologies by 2025:

- 680% increase in battery storage capacity (from 11MWh to 75MWh) [3]
- 170% increase in PV installation capacity (from 130MW to 220MW) [3]
- 39% of new car sales will be electrical vehicles - the highest in the country [4]

The changing nature of the distribution network presents an opportunity to maximize the use of existing assets by delaying the need for network augmentations, while also providing customers with a more reliable supply of power. For example, batteries could be used to peak-shift, reducing maximum feeder load. However, to achieve this reliably and with optimal use of available DER generally requires sophisticated methods to optimize the power flow to and from the distributed resources.

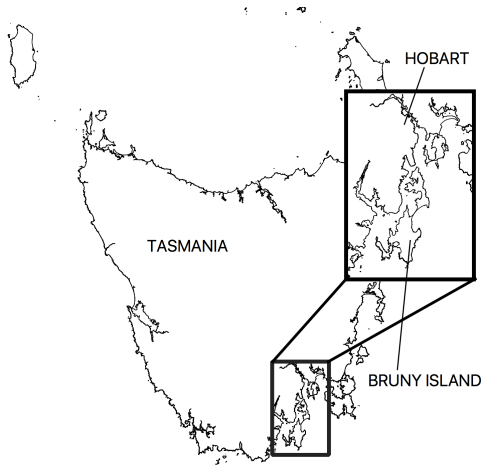

Fig. 1. Bruny Island in southeast Tasmania experiences significant increases in load during holiday periods and was used as a case study for the load forecaster.

One method to achieve this coordination of DER in distribution networks is presented in [5] and has been implemented on Bruny Island, Tasmania as part of the CONSORT project (CONSumer energy systems providing cost-effective grid suppORT). Bruny Island, shown in Figure 1, is a popular holiday destination and during peak periods — such as Easter morning and afternoon peaks — the submarine feeder supplying the island becomes overloaded and is supplemented by a diesel generator located on the island. The aim of the CONSORT project was to effectively peak shift the load away from the morning and afternoon peaks to prevent the use of the generator.

To fulfil such an objective while using the available distributed resources optimally, the CONSORT project relies upon having an accurate, online, 24-hour horizon forecast at the feeder level. However, load forecasting methods commonly employed in industry are neither intended to forecast with high accuracy over a time period this short nor at the feeder level. [6]. An improved method for producing accurate feeder-level forecasts is not only highly desirable for this project, but will also in the future become a critical element of active distribution network management more generally.

In this paper, a novel neural network-based day-ahead feeder level load forecasting system is developed, with its performance evaluated using ten years of historical demand data for training and testing. We implement the forecast live on Tasmania's Bruny Island distribution network, enabling the CONSORT project's residential battery systems to effectively support the network during periods of peak demand.

II. Forecasting System Architecture

Recurrent Neural Networks (RNN) have recently been popular for load forecasting in electricity networks [7]. However, RNNs have been out-performed by the Transformer [8] model in several domains including machine translation [8] (where the architecture was first proposed and applied), medical time series forecasting and regression [9], and image generation [10]. The Transformer is a purely attention-based neural network model and does not rely on recursion like traditional RNNs. This allows a larger degree of parallelism to reduce training time, while also allowing the model to effectively handle long-range dependencies in the inputs.

The proposed forecasting system uses a Transformer neural network model combined with similar period selection.

A. Similar Period Selection

Load profiles are influenced by exogenous data such as weather, day of the week, and holiday type [11]. A simple and intuitive method of load forecasting is to find periods in the past with similar exogenous data to the period being forecast and then use the load profiles from these past periods to form a forecast [12]. However, these similar period methods can be insufficient to capture complex patterns, especially over holiday periods which occur only once per year [13].

The forecasting system was provided with historical load and weather profiles from periods that had similar exogenous data to the period of the load profile being forecast. Similar periods were identified by first finding candidate similar periods an integer multiple of 1 year ± 30 days away from the period being forecast. These candidates were then filtered down to periods with exactly matching hour and minute.

Then the weighted Euclidean distance between the period being forecast and each candidate similar period was calculated using the following features: maximum future temperature, minimum future temperature, maximum past load, current holiday type, current day of the week, current day of the month, and current month of the year. The holiday type indicates the current holiday — Easter or Christmas for example — and is encoded as a time series of integers with a different integer for each holiday. When the holiday type always occurs on the same date each year then the month of the year and day of the month were used, whereas when the holiday type always occurs on the same day of the week each year then the day of the week was used. The candidate similar periods with the lowest Euclidean distance were selected as the final set of similar periods.

When training and testing the model the similar periods were selected from both the past and the future, as the train and test datasets were only five years each. It was assumed that, for testing, there were no changes in the patterns underlying the load profile over the duration of the testing set.

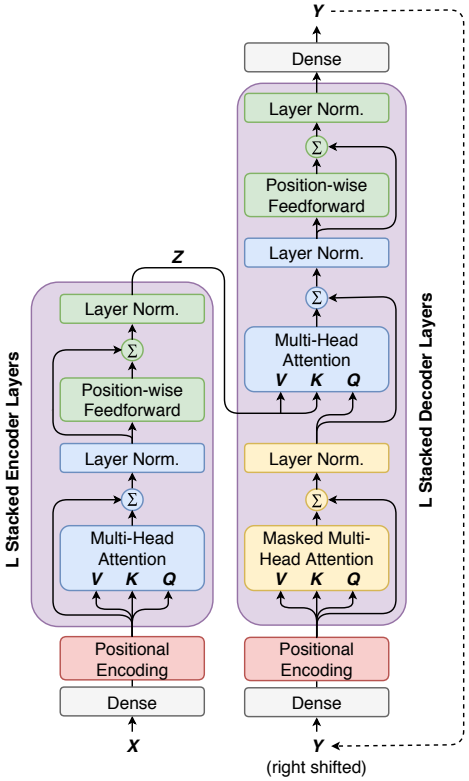

Fig. 2. The Transformer architecture.

B. Transformer

The transformer neural network architecture, shown in figure 2, was introduced by [8] in 2017 and at the time was the

state of the art in neural machine translation. This architecture follows the standard sequence-to-sequence/encoder-decoder architecture: the encoder transforms an input sequence $X = (\boldsymbol{x}_1, ..., \boldsymbol{x}_P)$ into a latent representation $Z = (\boldsymbol{z}_1, ..., \boldsymbol{z}_M)$, and the decoder transforms Z into an output sequence $Y = (\boldsymbol{y}_1, ..., \boldsymbol{y}_R)$, where x_t, y_t, and z_t are of arbitrary dimension. This matches the requirements of a load forecasting system, where P is the length of the time series input, x_t is of dimension equal to the number of variables in the input time series, R is the length of the time series output, and y_t is of dimension equal to the number of variables in the output time series. The latent representation Z is used solely by the model as an internal representation of the input data.

The encoder is constructed of a stack of L identical layers, each containing two sub-layers. The first is multi-head self-attention and the second is a position-wise feed-foward network. Both sub-layers have a residual connection around them and are fed into a normalization layer.

The decoder is similar to the encoder except for a third layer which implements multi-head attention on the outputs of the encoder. The input to the decoder is the previous output of the decoder, but shifted right by one. This requires an iterative approach to be used to predict all points in the time series. The self-attention in the decoder is masked so that when evaluating a query at time t it does not assign large weights to keys/values occurring after t in time, making the decoder autoregressive.

The individual components of the transformer are discussed in the following sections.

C. Input Embedding

The input $\boldsymbol{X} \in \mathbb{R}^{T \times N}$, where the rows represent T points in time and the columns represent N time series, is embedded by applying a dense layer to produce an embedded $\boldsymbol{Y} \in \mathbb{R}^{T \times d}$, where d is the hidden dimension of the model and d is the same for both the encoder and the decoder. This is intended to allow the neural network to learn the relationships and dependencies between the different input time series. The embedded representation is given by Equation 1, with learned weights $\boldsymbol{W} \in \mathbb{R}^{N \times d}$ and a learned bias vector $\boldsymbol{b} \in \mathbb{R}^{d}$.

$$\boldsymbol{Y} = \max(0, \boldsymbol{X}\boldsymbol{W} + \boldsymbol{b}) \tag{1}$$

D. Positional Encoding

The model has no way of telling the position or order of each element in the input, so this information is injected in the positional encoding layer. This is done by using a learned lookup table to add the same value to the inputs at both test and train time depending on their position in time in the input. Specifically, a matrix lookup table of embeddings $\boldsymbol{E} \in \mathbb{R}^{T \times d}$ is added to the embedded inputs as per Equation 2.

$$\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{E} \tag{2}$$

E. Multi-Head Attention

The primary innovation of the Transformer architecture is multi-head attention. Generic attention and dot-product attention will now be described as these are prerequisite to describing multi-head attention.

Given a single query vector and a set of key and value pairs (with each key and each value being a vector), an attention function matches the query to the keys to produce a weight for each key by applying an arbitrary fitness function. These key weights are then used to create an output vector comprised of the weighted sum of the values, where each value's weight is the weight assigned to its corresponding key.

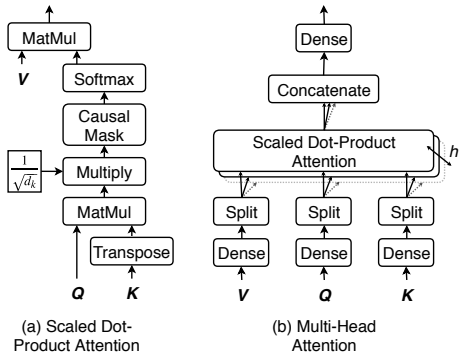

Fig. 3. Multiheaded attention (b) splits the key, query, and value matrices and applies scaled dot product attention (a) on each in parallel before concatenating the result to return the data to its original dimension.

Scaled dot-product attention, shown in Figure 3 (a), is a specific implementation of an attention function. It uses the dot product of the query and each key to generate the weights, which are then passed through a softmax function such that the sum of all weights is equal to 1. In practice, the query row vectors are combined into a single matrix, $\boldsymbol{Q}$, allowing cheap matrix calculations to be used to evaluate the attention outputs in parallel. The keys and values are represented by row vectors in $\boldsymbol{K}$ and $\boldsymbol{V}$, respectively.

The dot product of the keys and queries is scaled (hence the name) by multiplying it by $1/\sqrt{d_k}$, with d_k being the key and query dimension, to prevent the dot product from becoming large when d_k is large as this may cause the softmax gradient to become very small and affect the gradient descent training.

The causal mask shown in Figure 3 is used exclusively in the decoder self-attention to prevent the attention function from matching any query to a key that occurs after itself in time. This is achieved by leaving the lower triangular portion of the matrix untouched and setting the other values to be large negative numbers, indicating a very poor match.

Multi-head attention, shown in Figure 3 (b), applies a separate dense layer to each of the values, queries, and keys. The dense layer is applied per Equation 1 with learned weights $\boldsymbol{W} \in \mathbb{R}^{d \times d}$ and a learned bias vector $\boldsymbol{b} \in \mathbb{R}^{d}$. The outputs of the dense layers are then split along the last axis into h sets, or heads. As a result the key, query, and value dimension is reduced by a factor of h to $\frac{d}{h}$. Scaled dot-product attention is then run independently on each set. The results are concatenated and put through a final dense layer to produce the output of the attention function. The dense layer function

on the output is defined by Equation 1 where $\boldsymbol{W} \in \mathbb{R}^{d \times d}$ is a learned weight matrix and $\boldsymbol{b} \in \mathbb{R}^{d}$ is a learned bias vector.

The dense layer combined with the split allows the multi-head attention to pick out information from different subspaces in the input and direct these to different attention heads. This is in contrast to a single head which must average all subspaces.

F. Feed-forward

The feed-forward layer is a two layer network with a rectified linear unit in the middle. Given an input $\boldsymbol{X} \in \mathbb{R}^{T \times d}$, the output $\boldsymbol{Y} \in \mathbb{R}^{T \times d}$ is populated by Equation 3 where $\boldsymbol{W}_1 \in \mathbb{R}^{d \times 4d}$, $\boldsymbol{b}_1 \in \mathbb{R}^{4d}$, $\boldsymbol{W}_2 \in \mathbb{R}^{4d \times d}$, and $\boldsymbol{b}_2 \in \mathbb{R}^{d}$ are learned weights and biases.

$$\boldsymbol{Y} = \max(0, \boldsymbol{X}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2 \tag{3}$$

G. Decoder Dense Output

The output of the decoder is passed through a dense layer to project the hidden dimension to the desired dimension of 1. The layer is implemented per Equation 1 where $\boldsymbol{W} \in \mathbb{R}^{d \times 1}$ is a learned weight matrix and $\boldsymbol{b} \in \mathbb{R}^{1}$ is a learned bias vector. By adjusting the dimension of $\boldsymbol{W}$ and $\boldsymbol{b}$ the network could be modified to perform multiple forecasts simultaneously.

H. Residuals & Normalization

Residual connections [14] are applied around each sub-layer. That is, the output of each sub-layer is given by $\boldsymbol{Y} = \boldsymbol{X} + \text{subLayer}(\boldsymbol{X})$ where $\text{subLayer}(\boldsymbol{X})$ is the original output of the sub-layer. The outputs are then normalized by applying layer normalization [15], as per Equation 4 where $\mu_{\boldsymbol{x}}$ and $\sigma_{\boldsymbol{x}}$ are the mean and variance of $\boldsymbol{x}$ respectively.

$$\boldsymbol{Y}_{i,*} = \frac{\mu_{\boldsymbol{X}_{i,*}}}{\sigma_{\boldsymbol{X}_{i,*}}} \tag{4}$$

I. Dropout and Training

To help prevent overfitting to the training data, dropout [16] is applied during training at the output of both positional encoding layers, and immediately after the softmax operation in all multi-head attention layers. Additionally, the encoder inputs, decoder inputs, and expected outputs were summed with randomly distributed noise with a mean of 0 and standard deviation of 0.01 before being supplied to the model during training.

The model is trained using the Adam optimizer [17] and a modified sum of errors squared loss function. Given a vector $\boldsymbol{y}$ of predictions from the model and a vector $\boldsymbol{y'}$ of expected predictions the loss function l is given by

$$l = \sum_{t=0}^{R} ((\boldsymbol{y}_t - \boldsymbol{y'}_t)^2 \times |\boldsymbol{y'}_t|^c) \tag{5}$$

where c is a model hyperparameter. For $c > 0$ this function accentuates loss when the actual value is large — making the model more accurate at forecasting peaks.

When testing or being used for inference, the decoder outputs are generated sequentially one at a time. After each output value is generated it is shifted right by one and populated in the decoder input and the model is executed again until all the outputs have been generated. Values that have not yet been generated are set to zero in the decoder input. These zero values do not affect the output of the decoder, as the decoder self-attention is masked so that it does not make use of them. When training, the decoder input is set to the known expected value and the model is executed — and the learnable parameters updated — a single time.

III. Case Study

A. Bruny Island

Bruny Island, shown in Figure 1, is located approximately two kilometres off the coast of south-east Tasmania with a permanent resident population of approximately 800 people. The island is a popular holiday destination, with Easter periods typically experiencing an influx of up to 500 cars in a single day. The island is supplied by two feeders, depicted in Figure 4, with one feeder supplying a small portion of the island to the North and the other supplying the main portion of the island to the South. This case study deals only with the feeder supplying the main portion of the island.

During holiday period morning and afternoon peaks the submarine feeder reaches its capacity and a diesel generator located on the island is used to reduce the feeder load. The substantial increase in load over the Easter holiday period for multiple years can be seen in Figure 5.

To avoid the use of the generator, the CONSORT project installed a set of residential batteries on the island for the purposes of peak-shifting. These batteries are coordinated by the network aware coordination algorithm (NAC). In order to peak-shift while making efficient use of the batteries, the NAC requires an accurate forecast of load with a 24-hour horizon and 30-minute resolution.

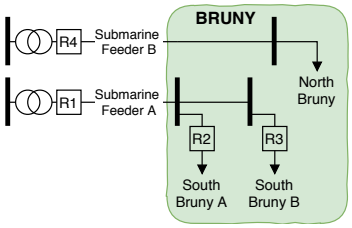

Fig. 4. Single line schematic of distribution network on Bruny Island. The majority of load is in the South of the island, fed from submarine feeder A, while a small load in the North is supplied by submarine feeder B.

B. Data and Model Configuration

The following data was available from 2009-2018:

- Apparent power at reclosers R1 through R4 (Figure 4).
- Temperature at Lenah Valley, Tasmania (50km from Bruny Island).
- Apparent power consumption at St Helens, Tasmania.

This data was averaged to 30 minute resolution and split into a training set containing data from October 2009 through September 2014, and a testing set containing data from October 2014 through April 2018.

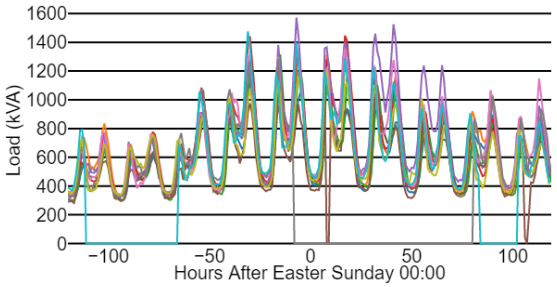

Fig. 5. Easter load on Bruny Island 2008 through 2017, showing the significant increase in load over this period when compared to the surrounding days. Unusable missing/bad data can also be seen in this graph.

The network was supplied with data from the previous and future 24 hours, for a total input sequence length of 96 (representing 48 hours at 30 minute resolution). The output sequence length was 48 (24 hours).

The following time series were supplied to the model input:

- Apparent power from recloser R1 (Figure 4), with future values set to zero.
- Temperature.
- Day of the week as an integer from 0 to 6 (local time).
- Minutes since midnight (local time).
- Boolean 1 or 0 indicating whether it is a holiday.
- Holiday type.

When used for inference, temperature forecasts were obtained from the Bureau of Meteorology.

Additionally, five similar periods were identified using data from R1 by the process described in section II-A. The data over the similar periods for each of the following time series was provided as input:

- Reclosers R1, R2, R3, and R4 (Figure 4) (as separate time series).
- St Helens recloser.
- Lenah Valley temperature.

In total, 36 time series were provided as input to the model. St Helens was included because it was observed to display similar patterns to Bruny Island around holiday periods.

The forecasting system was configured with the parameters in table I, with the upper section giving transformer model parameters and the lower section giving weights used for similar period selection.

C. Results

The forecaster was first evaluated on historical data around Easter 2018, shown in Figure 6. Notably, the forecaster was able to accurately predict the first large peak. This is in contrast to load forecasting models which sometimes tend toward trivially repeating the previous day's load.

The performance of the forecaster was evaluated on every Easter, Queen's Birthday, and July school holiday period from 2015 through 2018 (2018 excludes July). The results are shown in Figure 7, showing the mean absolute percentage error (MAPE) as a function of forecast horizon. The mean MAPE is 7.4%. Furthermore, the errors between predicted and actual load are fairly evenly distributed around -15 kVA, shown in Figure 8. This indicates that the model has some room for improvement when predicting large holiday peaks, but overall this is evidence that the model has been able to generalize from the training data, as the training data is mostly comprised of normal days.

When implemented live on the Bruny Island distribution network, during the July 2018 school holiday period, the forecaster was observed to reliably forecast large demand peaks. This enabled the fleet of distributed batteries to be used effectively in providing network support via net demand peak reduction. An accurate forecast, issued early enough in

TABLE I
CASE STUDY MODEL PARAMETERS.

Parameter	Description	Value
L	Number of encoder and decoder layers	4
d	Hidden dimension	32
h	Number of attention heads	4
D	Dropout fraction	0.2
c	Loss function modifier	3
-	Training batch size	16
-	Maximum future temperature weight	10
-	Minimum future temperature weight	20
-	Maximum past load weight	30
-	Holiday type weight	1e9
-	Day of week weight	1e6
-	Day of month weight	1e6
-	Month of year weight	1e6

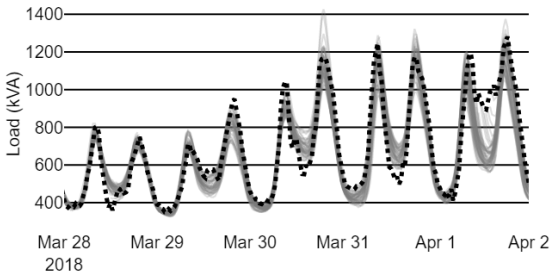

Fig. 6. Forecasts over the Easter 2018 period. The black dashed line is the actual recorded load, and all previous forecasts are shown in grey. The single system is able to transition smoothly between normal and holiday periods.

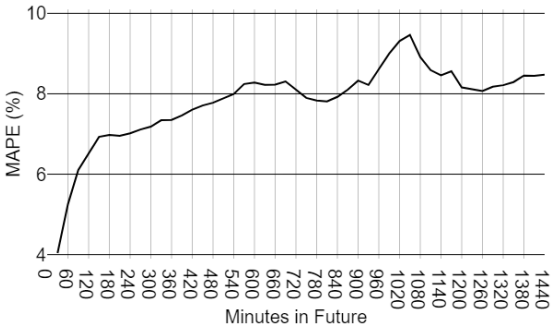

Fig. 7. Mean absolute percentage error of each point in the forecast when evaluated on every Easter, Queen's Birthday, and July school holiday period from 2015 through 2018 (2018 excludes July). The mean MAPE is 7.4%.

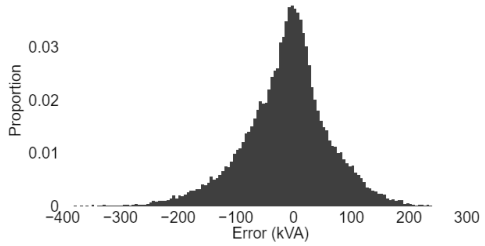

Fig. 8. Distribution of forecast error when evaluated over the same periods as Figure 7. The error has a mean of -15 kVA.

advance of the occurence of the demand peak, was observed to give the batteries adequate time to store energy in the lead up to, and discharge during the demand peak period. In at least one instance over the test period this was sufficient to avoid the island's diesel generator from being used at all, when it otherwise almost certainly would have been required. Data collected during this peak demand period can be seen in Figure 9. The upper section shows 24 hours of historical load in black, plus the most recent 24-hour horizon forecast in dashed black (recalculated every five minutes) and all old forecasts in grey. The lower section shows the battery charge rate, where a negative value of battery charge rate indicates the batteries are supporting the grid.

Typically the generator is switched on when load exceeds 1050 kVA. During the first peak the graph shows the batteries supplying between 50 and 100 kW to the island. Without this support from the batteries, the generator would have been required to operate.

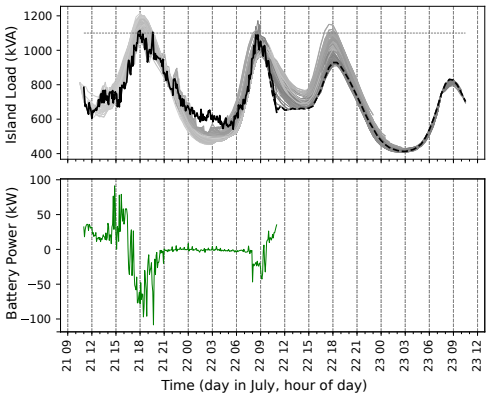

Fig. 9. The forecaster was able to predict the major peak (top, 21 July 18:00), enabling the batteries, rather than the generator, to support the island (bottom).

IV. Conclusion

This paper presents a novel neural network-based load forecasting system and applies it to Bruny Island, Tasmania, allowing a fleet of residential batteries to effectively support the network during major peaks. The single load forecasting system was able to accurately predict peaks both during anomalous holiday periods and during periods of normal load, with a mean MAPE of 7.4% and a mean error of -15kVA. It is expected that this system would be equally applicable to any distribution feeder, and could be trivially expanded to perform multiple forecasts simultaneously by adjusting the output dimension of the final dense layer of the neural network.

Acknowledgment

This work has been supported by TasNetworks, who also provided network and demand data. It has also been supported by researchers from the ARENA-funded CONSORT Bruny Island Battery Trial.

References

[1] W. Johnston, "National survey report of photovoltaic applications in australia 2017," 2017. [Online]. Available: http://apvi.org.au/pv-in-australia-2017/

[2] Bloomberg New Energy Finance, "New energy outlook 2018," 2018. [Online]. Available: https://about.bnef.com/new-energy-outlook/

[3] W. Gerardi and D. O'Connor, "Projections of uptake of small-scale systems," 2017. [Online]. Available: https://www.aemo.com.au/Electricity/National-Electricity-Market-NEM/Planning-and-forecasting/Electricity-Forecasting-Insights/2017-Electricity-Forecasting-Insights/Key-component-consumption-forecasts/PV-and-storage

[4] "2016 national electricity forecasting report," 2016. [Online]. Available: https://www.aemo.com.au/Electricity/National-Electricity-Market-NEM/Planning-and-forecasting/National-Electricity-Forecasting-Report

[5] P. Scott and S. Thiébaux, "Distributed Multi-Period optimal power flow for demand response in microgrids," in *ACM e-Energy*, Bangalore India, jul 2015. [Online]. Available: http://users.cecs.anu.edu.au/~pscott/extras/papers/scott2015.pdf

[6] CIGRE. (2016) Cigre working group concludes demand forecasting study. [Online]. Available: https://www.cigreaustralia.org.au/news/features/cigre-working-group-concludes-demand-forecasting-study/

[7] W. Kong, Z. Y. Dong, D. J. Hill, F. Luo, and Y. Xu, "Short-term residential load forecasting based on resident behaviour learning," *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 1087–1088, jan 2018.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5998–6008. [Online]. Available: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[9] H. Song, D. Rajan, J. Thiagarajan, and A. Spanias, "Attend and diagnose: Clinical time series analysis using attention models," 2018. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16325

[10] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, and A. Ku, "Image transformer," *CoRR*, vol. abs/1802.05751, 2018. [Online]. Available: http://arxiv.org/abs/1802.05751

[11] R. Weron, *Modeling and Forecasting Electricity Loads and Prices*. John Wiley & Sons Ltd, dec 2006.

[12] T. Senjyu, S. Higa, and K. Uezato, "Future load curve shaping based on similarity using fuzzy logic approach," *IEE Proceedings - Generation, Transmission and Distribution*, vol. 145, no. 4, p. 375, 1998.

[13] Y. Chen, P. Luh, C. Guan, Y. Zhao, L. Michel, M. Coolbeth, P. Friedland, and S. Rourke, "Short-term load forecasting: Similar day-based wavelet neural networks," *IEEE Transactions on Power Systems*, vol. 25, no. 1, pp. 322–330, feb 2010.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[15] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[16] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2627435.2670313

[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.