Imperial College London

Neural NILM: Deep Neural Networks Applied to Energy Disaggregation

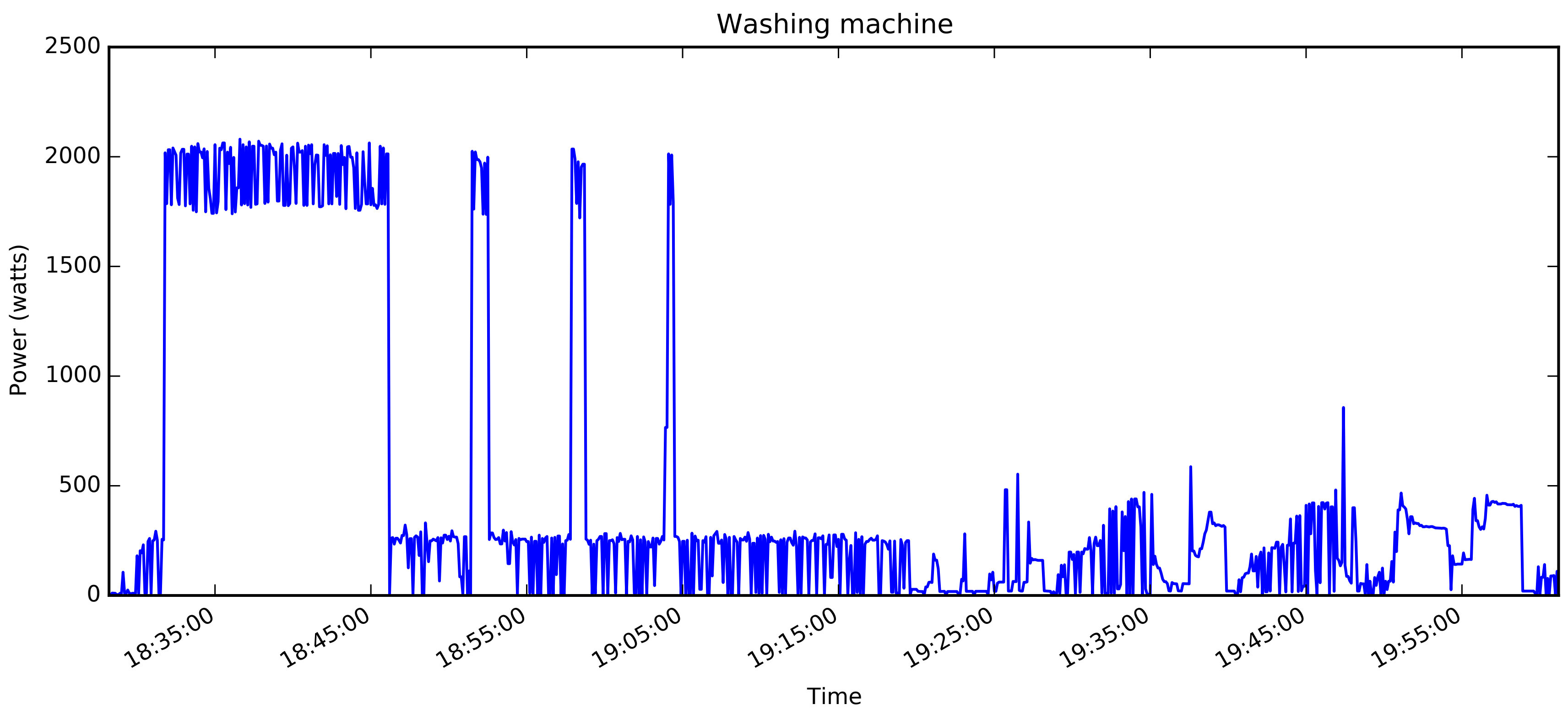
Jack Kelly &
William Knottenbelt
jack.kelly@imperial.ac.uk
Department of Computing, Imperial College London

1) INTRODUCTION

Artificial Neural Nets have gone in and out of fashion at least twice since their birth in the 1950s. In the last few years, 'deep' neural nets have achieved state of the art performance on many machine learning tasks including image classification, automatic speech recognition, handwriting recognition, machine translation, learning Atari games (!) etc. In many of these tasks, deep neural nets now have a substantial performance advantage over other approaches.

Automatic Feature Learning: 'Classical' machine learning often involves hand-engineering a number of feature detectors. These hand-engineered feature detectors are time consuming to engineer and are often fragile in practice. Deep neural nets don't require hand-engineered feature detectors. Instead they *learn* a hierarchy of feature detectors from the data. Given enough training data, deep neural nets learn feature detectors which are often more robust and more informative than hand-engineered feature detectors.

Feature Detectors for NILM: Many existing NILM algorithms extract a small number of features from the aggregate power signal. But if we train *ourselves* to identify appliances, we use a rich set of features. For example, a washing machine's drum motor regularly stops and starts during the rinse cycle. These rapid spikes in the power demand signal are a clear indication of a washing machine. But most NILM feature extractors ignore these spikes as 'noise'. We wanted to see if deep neural nets might be able to learn to detect these rich features.



2) METHOD

Network input: We train 1 net per target appliance. The network input during training and inference is a window of raw *aggregate* time series data (after gaps have been filled). The window length is set to be a little longer than the maximum duration of an appliance activation (e.g. 2 hours for a washing machine or 5 minutes for a kettle).

Network output: The network tries to locate the target appliance and its mean power demand by drawing a rectangle over the target appliance in the aggregate input. The network outputs 3 positive real numbers: 1) the start time of the target appliance activation, 2) the end time and 3) the mean appliance power demand over the activation.

Network layers:
1 = Input layer varies in size per appliance
2 = 1D convolutional layer (number of filters=16, filter size=4, stride=1)
3 = 1D convolutional layer (number of filters=16, filter size=4, stride=1)
4 = Dense layer (8192 rectified linear units)
5 = Dense layer (4096 rectified linear units)
6 = Dense layer (2048 rectified linear units)
7 = Dense layer (512 rectified linear units)
8 = Output layer (3 linear units)

We experimented with dropout and batch normalisation but neither appeared to add anything useful.

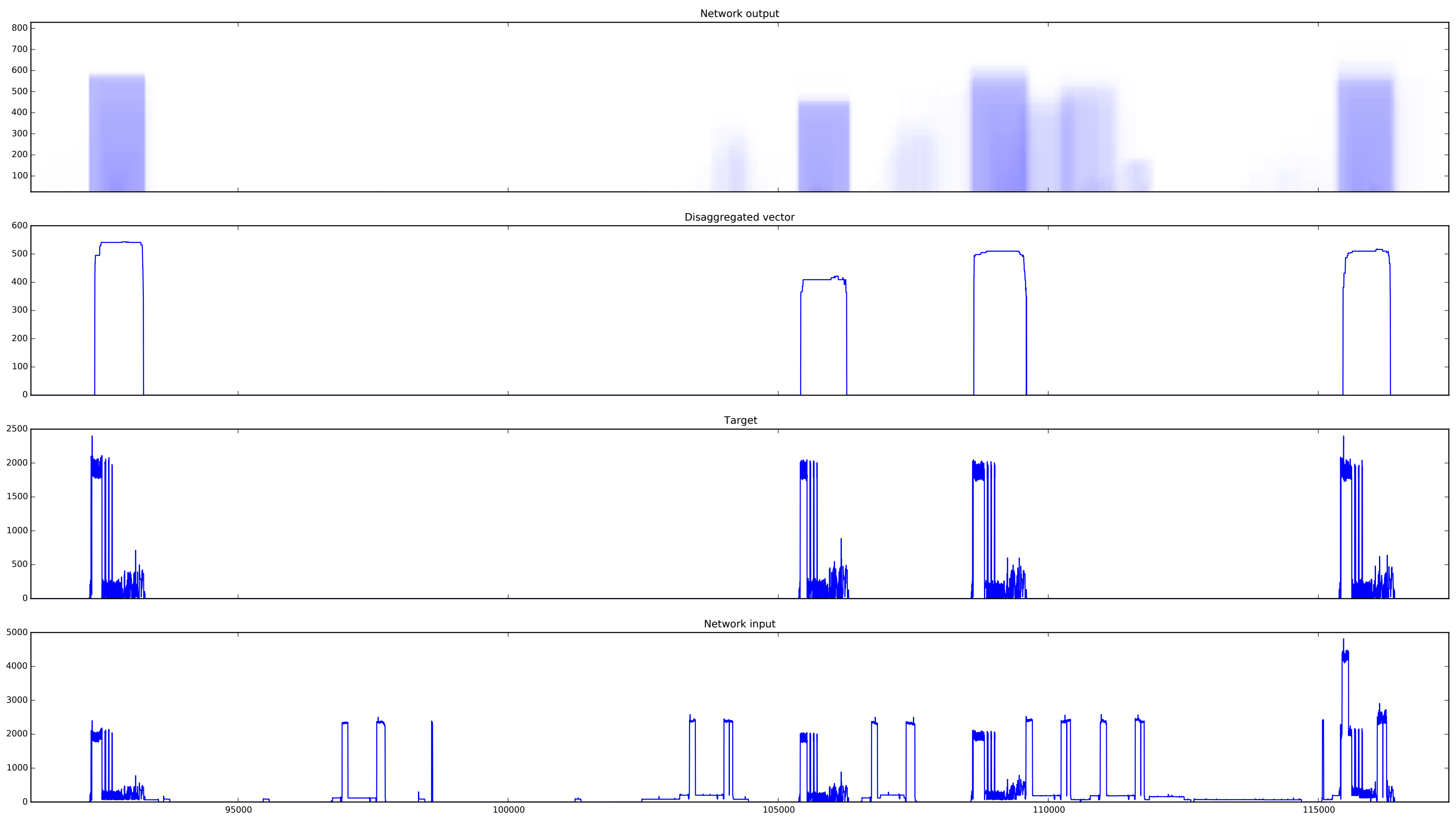
Training: Deep neural nets require a *lot* of training data. So we generate an effectively infinite amount of training data by randomly combining real appliance activations into synthetic aggregates. The training objective is to minimise MSE. We trained on an nVidia GTX780Ti GPU for ~24 hours per appliance. We use the Python library Lasagne (built on top of Theano).

Inference: How do we cope with arbitrarily long sequences given that each net has an input window duration of 2 hours or shorter? We slide the net along the input sequence (with stride = 16). We then layer every predicted 'appliance rectangle' on top of each other. We measure the overlap and normalise the overlap to [0, 1]. This gives a probabilistic output for each appliance's power demand. To convert this to a single vector per appliance, we threshold the power & probability.

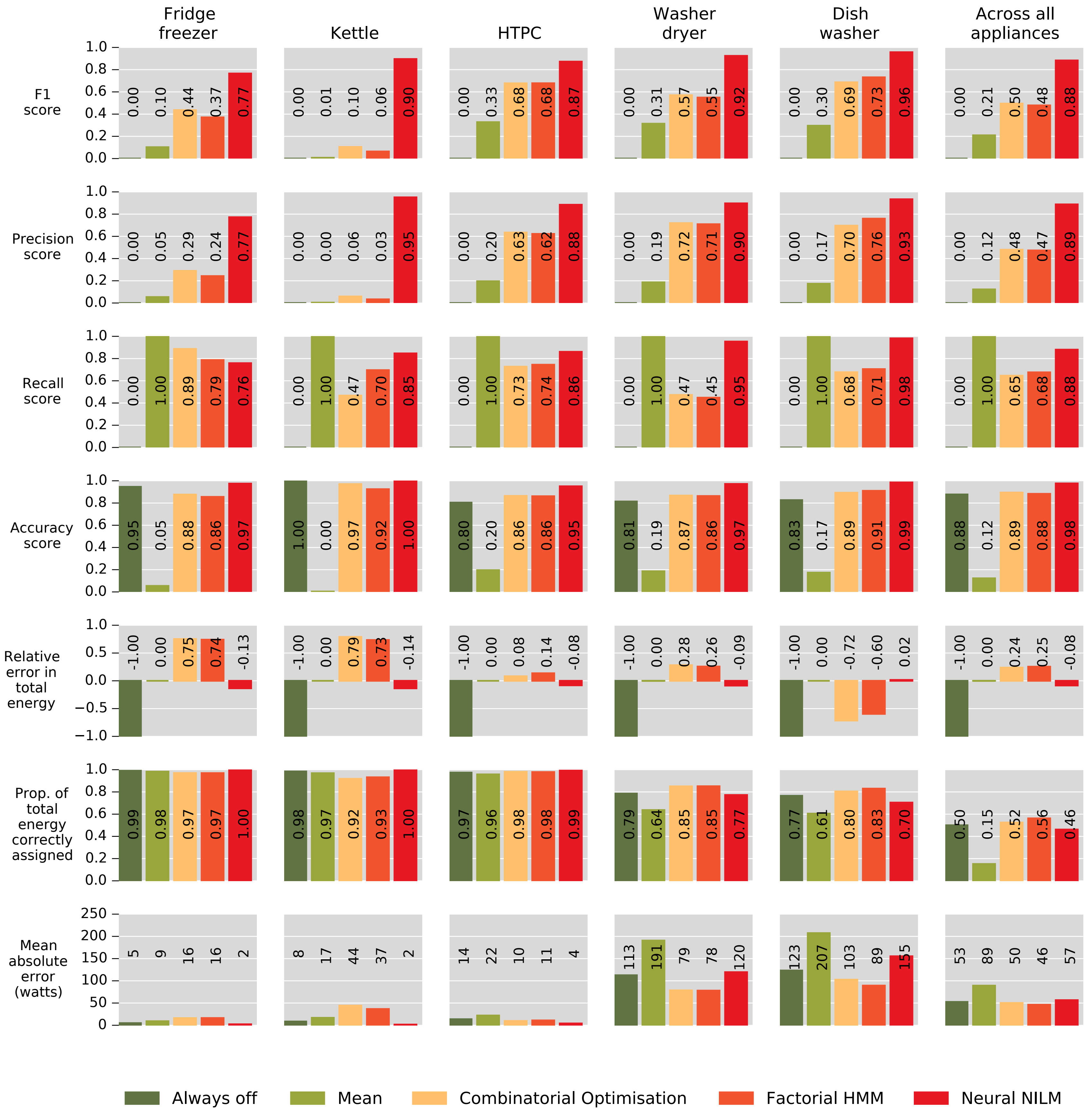
3) RESULTS

We generated 9 days of synthetic aggregate data using appliance activations not seen by the nets during training.

Raw network output: The plot below shows, from the bottom panel up: 1) a window of synthetic aggregate; 2) the ground truth target (4 activations of a washing machine); 3) the net's vector output; 4) the probabilistic net output (overlapping rectangles).



Comparison with other NILM algorithms: we trained and tested two benchmark algorithms from NILMTK[2] on the same data that we used with Neural NILM. For comparison, we also included two 'dumb' algorithms: one which always outputs zeros (!) and one which always outputs the mean. Neural NILM out-performs the other algorithms on every metric but one. Neural NILM doesn't come first on the 'proportion of total energy correctly assigned across all appliances' because Neural NILM doesn't try to track state changes on multi-state appliances like the washer dryer and dish washer. But this should be fixable in a future version of Neural NILM!



recall = true positive rate = sensitivity = TP / P = TP / (TP + FN)
precision = positive predictive value = TP / (TP + FP)
accuracy = (TP + TN) / (P + N)
relative error in total energy = |total predicted energy - total actual energy| / max(predicted, actual)

Proportion of total energy correctly assigned $= 1 - \frac{\sum_{t=1}^{T}\sum_{i=1}^{n}\left|\hat{y}_t^{(i)} - y_t^{(i)}\right|}{2\sum_{t=1}^{T}\bar{y}_t}$

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

4) CONCLUSIONS AND FUTURE WORK

We have demonstrated the first application of DNNs to NILM. This first version of Neural NILM already out-performs CO and FHMM algorithms. There are a large number of improvements we are planning to try.

REFERENCES

[1] Jack Kelly & William Knottenbelt. The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. Scientific Data 2, Article number:150007, 2015. DOI:10.1038/sdata.2015.7
[2] Jack Kelly, Nipun Batra, Oliver Parson, Haimonti Dutta, William Knottenbelt, Alex Rogers, Amarjeet Singh, Mani Srivastava. Demo Abstract: NILMTK v0.2: A Non-intrusive Load Monitoring Toolkit for Large Scale Data Sets. In the first ACM Workshop On Embedded Systems For Energy-Efficient Buildings, 2014. DOI:10.1145/2674061.2675024

ACKNOWLEDGEMENTS

This work was funded by a EPSRC DTA and an Intel PhD Fellowship.