9th International Conference on Prestigious Applications of Intelligent Systems – PAIS@ECAI2020
Santiago de Compostela, Spain

Deep learning applied to NILM: is data augmentation worth for energy disaggregation?

Aurélien DELFOSSE, Georges HEBRAIL, Aimen ZERROUG [1]

Abstract. Energy disaggregation (or Non-Intrusive Load Monitoring - NILM) is the task of estimating the electricity consumption of each appliance in a household from the total electricity consumption. Disaggregated consumption gives information on each appliance and helps to find ways to reduce a household's energy consumption. Recent progress in deep neural networks for computer vision and natural language processing gives inspiration to train general architectures on time series data in order to improve the state of the art on NILM, but lack of supervised data is one of the main problems stalling the improvement of disaggregation algorithms. In this paper, we introduce a new multi-agent based simulator that enables to generate synthetic data according to real time-use surveys. This synthetic dataset is used as a training set in the NILM learning process: we show that this data augmentation improves the accuracy of the disaggregation. In addition, we present four neural network architectures to estimate appliances consumption and establish a baseline architecture on the data coming from the synthetic generator.

1 INTRODUCTION

Non-Intrusive Load Monitoring (NILM) or energy disaggregation is the process of estimating the consumption of each individual appliance from the global consumption of a household. The widespread installation of smart meters in individual houses gives an opportunity for using such a technique to help users monitor and reduce their energy consumption. The first application is to inform the occupant of the amount of energy each appliance consumes. A second application is to provide a personalized feedback in case of an appliance's malfunction or inefficiency to prevent breakdowns. Finally, we can warn the household's occupant of the potential savings of differing appliance use to a time of the day when electricity is cheaper or has a lower carbon footprint. The last application can be associated with solar panel system, in order to maximize the rate of self consumption electricity.

The NILM problem, which was formalized in the mid-1980s by George Hart [5], was focused on extracting transitions between steady-states. Many NILM algorithms on low frequency data (1 Hz of lower) followed Hart's lead and extracted only a small number of expert features, followed by a classifier like support vector machine (SVM) or K-Nearest-Neighbour(KNN). These handcrafted feature extractors are not used anymore since deep neural networks can automatically learn to extract a hierarchy of features in various objects like raw images [17], texts [6] or time series. Deep learning for NILM was introduced in 2015 by Jack Kelly [9] with major progress on state of the art models, and made a major breakthrough against old methodology, using three different architectures for time series classification or regression.

The difficulty of NILM lies in the fact that several appliances, which are not always present in the house, can be used and this forces us to train several models to make multi-output prediction. Many NILM algorithms have been designed for high frequency data (sampling at kHz or even MHz) to low frequency data (1Hz- or slower) like the REDD, blueED or UK Dale dataset [10] [8] sampled from 1 to 6 seconds. These public datasets contain both aggregate and submetered energy data for each monitored house, but are not easily transferable to another sampling rate or country with another energy consumption behaviour. For example, almost all datasets will not be reusable with the expected integration of electrical vehicles around the world and new behaviour with self electricity generation. In our experimentation, faced to the lack of 2 second data samples, which will be the new standard for downstream smart metering in France, we introduce a Multi-Agent based Simulator (MAS), SMACH [15], which uses real appliances signatures to generate a synthetic dataset that follows the distribution of real data. This data augmentation trick allows us to train networks that generalize over the real datasets and unseen houses. French data is particularly complex due to the variety of heating appliances, this makes disaggregation an even harder challenge. The MAS is flexible enough to generalize over other countries or behaviour not seen yet, and avoids data collection campaigns to integrate significant changes in behaviour.

Since neural networks training is computationally intensive, we use modern GPU and multi-nodes HPC environment to train several deep learning architectures adapted to time series prediction.

In Section 2, we briefly review the state of the art of the NILM problem. In Section 3, we describe the data augmentation process to generate synthetic data used for training. Section 4 presents the experimental setup for one particular appliance, the water heater, which is one of the most consuming appliance. This section describes the different generated datasets and the cross validation methodology used to assess the intake of the simulator and its ability to generalize, as well as the metrics. We detail the architecture choices and their performance on water heater consumption prediction using 2 metrics. Section 5 presents the disaggregation results for three neural network models that were trained on synthetic data and tested on real data.

2 NILM BACKGROUND

Hart's [5] work focused on extracting transitions between steady-state considering a small number of features and used combinatorial optimisation to find the optimal combination of appliance states which minimises the difference between the sum of the predicted

[1] ELECTRICITE DE FRANCE EDF Lab Saclay, France, email: firstname.lastname@edf.fr

appliance power and the observed aggregate power. The second historical approach was proposed in 2013, and consist in using Factorial HMM (FHMM) [19], [2]. The power demand of each appliance can be modelled as the observed value of a hidden markov model. The hidden component of the HMMs are the states of the appliances. FHMM can be viewed as an HMM where each state corresponds to a different combination of states of each appliance. Both of these algorithms are bad at scaling, with a high complexity. Beyond these methodologies, several improvements have been made to automatically extract hierarchical sets of features. A breakthrough in supervised learning was made in 1989 when LeCun successfully applied neural network and back propagation to handwritten digit recognition [17]. The fully connected neural network pipeline actually learns automatically relevant features and makes the prediction at the same time. This kind of approach took almost 20 years to become the dominant algorithm thanks to the victory in 2012 in the ImageNet Large Scale Visual Recognition Challenge of a deep Learning approach [3] and the use of GPU (graphical processor units) which increases drastically computer power computation. Since 2012, deep learning in most of supervised learning is considered as the state of the art approach in many complex problems, owing to its ability to learn deep representations of the data, and build complex parametric functions. This flexibility has made this method a baseline approach in computer vision [3], natural language processing [6], time series classication [16] or event audio synthesis [1]. Logically, deep learning becomes unavoidable to solve the NILM problem. In 2015, [9] used three neural network architectures (convolutional network, denoising autoencoder, start&end time regressor) to predict the activation or consumption of several appliances. Using a mix of the UK-DALE [10] dataset sampled at 6 seconds and synthetic data, Kelly outperformed preceding methods like combinatorial optimisation and factorial hidden Markov models [2], [5]. In his experiments, he generated synthetic data by randomly combining appliance signatures. The selected appliance must fit entirely in the window presented to the neural network and is selected with a probability of 0.5. He raised the point that a realistic simulator that respects temporal structure of appliance's activations might increase the performance of deep neural nets compared to this naïve approach which doesn't consider this temporal structure.

Several papers were released since 2012, suggesting various approaches for disaggregation. [4] propose a Seq2Seq and a Seq2point method, learning the mid-point of sliding windows based on an input short sequence, instead of the 14,400 time steps signal. This approach has the advantage to manage smaller window sizes, and avoids the gradient vanishing problem [14] during the training phase. Most of recent approaches use a LTSM layer [11],[18], which is more suitable for long sequence, using input, output and forget gate to manage memory and gradient backpropagation. Finally, [13] propose to predict the operational state change of appliances, using a smaller neural network architecture and a power threshold value to determine if the appliance is ON/OFF.

Since sequence modeling is still a opened challenge in machine learning, an active community is trying to evaluate the best architecture depending on the type of data [16] and has made several improvements since Kelly's paper. In this paper, we evaluate some of the most promising RNN architectures adapted to long time series regression.

3 DATA AUGMENTATION APPROACH

3.1 Multi-agent based simulations of human activity

Gathering training data for NILM is very expensive because it requires many volunteer households to monitor their energy consumption with meters on several appliances. Indeed, for every household, we need both the aggregated and each sub-metered appliance to build the training dataset. We encounter three main difficulties in such data collection: (1) the presence of a many different appliances (oven, washing machine, fridge..) which are not always available in each house; (2) the high variability between appliances signatures, both in terms of amplitude and pattern that we can meet on the appliance market ; (3) the behaviour of the household occupant, which is pretty unstable according to the position and the composition of the house.

With a constantly changing energy market, the difficulty to collect enough real aggregated and sub-metered data, and the necessity to study electrical vehicle consumption and develop personalized services, the need for simulated data is on the rise. Facing this challenge, EDF R&D developed a multi-agent simulator (MAS), which is flexible enough to adapt to new uses. Quentin Reynaud and Yvon Haradji [15] proved that the integration of real *time use surveys (TUS)* in the simulator improves the realism of simulated individual behaviors. The original TUS used in the simulator is based on 10 minutes activity reports (27000 reports in total) which are representative of the french population's behaviour. It helps to establish a sequence of actions for each profile in the MAS, defined by its duration, rhythm and preferential period of use. In each TUS is taken in account the localisation, the composition of the household and the type of pricing. These information are valuable in order to represent all the complexity of the electrical consumption behaviour. This result in more realistic scenarios. The output of the MAS is a sequence of activation for each available appliances, that is adapted according to the composition and localisation of the household. Moreover, for each generated scenario, time of the year, city and household composition are available in order to use it as additional features in the model. The simulator is flexible enough to integrate new usages like electrical vehicle, in addition to eleven appliances like kitchen appliances, heater, vacuum cleaner, water heater or washing machine. Output scenario can be described as a table in which each column is an appliance, and rows correspond to time steps. For each time step, each column indicates if the appliance is activated (1) or not (0).
The final file we consider in this study has the following format (below is the exhaustive list of each available appliances):

1. Time : Timestamp index (2sec).
2. vacuum cleaner : 1/0 (activate or not).
3. water heater : 1/0.
4. washing machine : 1/0.
5. dishwasher : 1/0.
6. light : 1/0.
7. fridge : 1/0.
8. TV : 1/0.
9. oven : 1/0.
10. computer : 1/0.
11. electrical vehicle : 1/0

3.2 Synthetic training data generator

We use weekly MAS simulated scenarios to build a 2 seconds re-sampled synthetic dataset with the help of real appliances signatures (apparent power). Appliance signatures are extracted from real data collected from 27 houses during our own experimentation called *electric data*. In comparison, the REDD dataset [8] published by the MIT uses data coming from 6 houses. The experimentation enabled us to gather aggregated and sub-metered load curves of a large variety of appliances. An adaptation of the *get_activation()* function created by Kelly that is available in the NILMTK [12] was used to extract individual appliance signatures from the real dataset. We adapted the threshold value to extract signatures specifically for french appliances. Data generation is performed using 10000 weekly scenarios with 10 minutes time steps, and a database of extracted signatures. These weekly scenarios are first transformed into 2 seconds time series by repeating 0/1 appliance activation values every 2 seconds within the 10 minutes activation/deactivation periods. For each appliance type in a generated scenario, we select a random appliance and then copy random signatures from this same appliance during the scenario when the appliance is activated. Notice that in a generated scenario, the different appliances can come from different houses, providing use diversity in the final dataset. This allows us to use the same scenario multiple times and obtain a different generated output, but with a preserved temporal structure also ensuring privacy. Finally, we sum all individual appliance consumptions to create the aggregated load curve of the household and export the result in HDF5 format.

Below, we present the pseudo-code for data generation based on a MAS generated scenario s. We note:

- n the time length of a scenario ($t = 1..n$)
- $signature(i, j, k)$ the signature k ($k = 1..n_{i,j}$) of appliance j ($j = 1..n_i$) of type i ($i = 1..10$)
- $load_i$ the disaggregated load curve of appliance of type i, $load_{agg}$ the aggregated load curve

For each generated scenario s, we define the procedure to simulate aggregated and disaggregated load curves:

1. Select scenario s
2. For each type of appliance i appearing in s
   - Choose randomly an appliance j of type i
   - Initialize a vector $load_i$ of size n filled with zero values
   - For each time step t which is the beginning of an activation of type of appliance i
     - Choose a random k in $1..n_{i,j}$
     - Copy $signature(i, j, k)$ to $load_i$ from time step t to $t + length(signature(i, j, k))$ while the appliance of type i remains active
3. Compute aggregate curve $load_{agg} = \sum_i load_i$

Note that if one wants to apply this approach to another area, it is important to use a time use survey and appliance signatures from that country because electricity usage varies significantly between location. As a matter of fact, different countries use different sets of appliances and there are different usage patterns between cultures.

Our main contribution in this publication is to highlight the need for a multi-agent simulator to generate synthetic training data and evaluate the interest on state of the art modeling on long time series prediction.

4 EXPERIMENTAL SETUP

4.1 Data

In order to fit a baseline neural network model, we reshape the 10000 weekly 2 seconds generated load curves to a daily shape, which gives us a dataset of 70000 load curves with an input dimension of 43200 time steps.

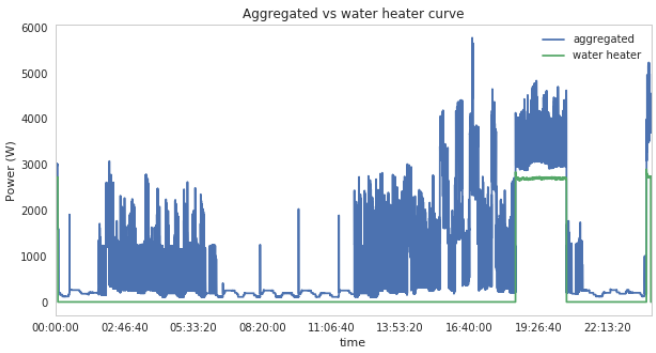

Figure 1. Example of generated aggregated and water heater curve (1 day - 2 seconds sampling rate).

Each model is evaluated on a real dataset composed of 1300 daily load curves coming from 10 houses for the water heater. The target to predict is the daily consumption of the appliance in kWh (kilo watt per hour). This is a regression task.

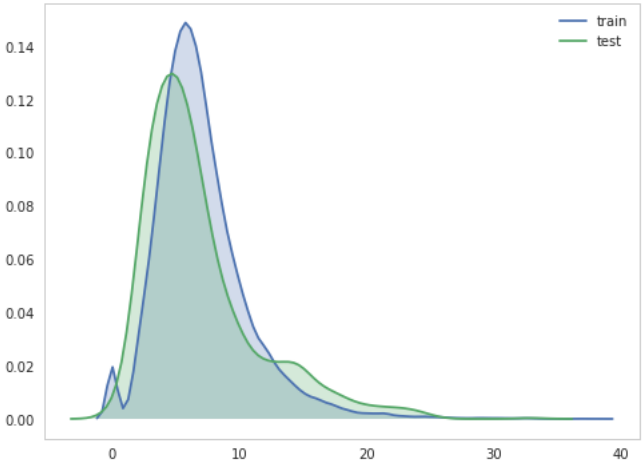

Figure 2. Distribution of train/test water heater consumption in kWh.

4.2 Data preparation for cross validation experiments

In order to evaluate the performance of our regressor on the prediction of consumption based on the daily consumption curve, we generate 10 training datasets. Each dataset excludes the signatures of one house: the excluded house signatures are then used for creating

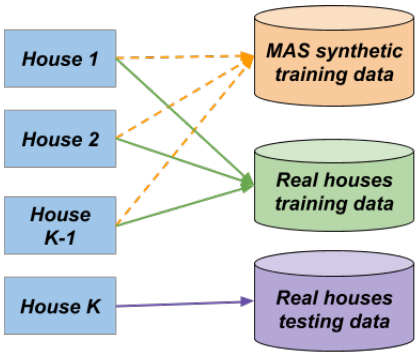

Figure 3. Data source for train and test generation.

the corresponding test set. Excluding the test house's signatures from the generated training dataset is important if we don't want to be too optimistic and assess the ability to generalize on unseen houses. We train 10 models (one for each training dataset) and predict the house excluded from the dataset to compute the final performance.

The goal of these experiments is to determine whether a synthetic dataset can replace a real dataset and test the baseline architecture capability of generalizing over curves with consumption signatures that were not used for training.

We first define the following training and test datasets:

- $D_1, D_2, ...D_{10}$ are the datasets of real real data corresponding to each of the 10 houses and D is the union of these datasets.
- If we choose house $k = 1..10$ as the house selected for testing, we note:
  - $D_{test} = D_k$ the test set (purple in Figure 3)
  - $D_{real} = D - D_k$ the training dataset containing real data of all houses except house k (green in Figure 3)
  - D_{synth} the synthetic dataset obtained with the MAS approach using signatures from all houses except house k (orange in Figure 3)

Then we define the different training and test setups:

- **Baseline Learning**: train a model M_1 on D_{synth} and test on D_{test}.
- **Transfer Learning FL (Full Layers)**: fine tune all layers of M_1 on D_{real} and test on D_{test}.
- **Transfer Learning 2L (2-Layers)**: fine tune the 2 last layers of M_1 on D_{real} and test on D_{test}.
- **Full Learning**: train a model M_2 on the union of D_{synth} and D_{real} and test it on D_{test}.
- **Subset learning**: train a model M_3 on D_{real} and test on D_{test}. The training is slightly unstable due to the small size of the training dataset.

4.3 Standardization

Individual energy consumption data is generally skewed to zero, thus training is less stable if we use raw input data. In time series classification or regression, preprocessing is an important task as it influences the way of learning. An instance preprocessing based on local mean will focus on shape, whereas a global normalization will favor the level of the curve. To standardize input, we first use the hyperbolic $arcsinh$ with a parameter α which unskews the data and makes statitics of the data less sensitive to outliers, then we normalize the result. Normalization is then just a rescaling which speeds up training as follows:

$$X' = \frac{arcsinh(\alpha * X)}{\alpha} \quad (1)$$

$$X'' = \frac{X' - \mu_{X'}}{\sigma_{X'}} \quad (2)$$

Mean and standard deviation of the training set is used to standardize the test dataset.

4.4 Neural networks architectures

We have tested two main architectures inspired from sequence modelling, time series [20] [7] and signal analysis prediction. The best architecture was then used to evaluate our model's cross validation performance as explained in 4.2.

Likewise [4], we choose to model the problem with a SeqToPoint and a SeqToSeq architecture. Suppose that we are given an input sequence $(x_0, x_1, .., x_T)$, and wish to predict some corresponding outputs $(\hat{y}_0^i, .., \hat{y}_T^i)$ at each time. The Seq2Seq model consists in predicting $\hat{y}_t^i$, that represent the estimated consumption of appliance i at time t. The SeqToPoint architecture focuses on estimating $\hat{y}^i$ which is the consumption of the whole input.

The two considered neural network architectures are the following:

- Time Distributed convolutional networks: we reshape the input from a (43200) shape to a sequence $(48, 900)$ of 48 timestamps. This approach was successfully implemented by the University of Tianjin [7] to estimate machine health from a multi-sensory input. We use this architecture both for SeqToPoint and SeqToSeq model. The prediction of the second one is calculated by summing all the predictions. It means that based on the model $(\hat{y}_0^i, .., \hat{y}_{48}^i) = F(x_0, x_1, .., x_{48})$, we have $\hat{y}^i = \sum_{t=1}^{48} \hat{y_t^i}$

  The ConvBlock is built with 64 filters, a kernel size of 5, batch normalization and max pooling of size 2. The activation function is *ReLU* for all layers. The *Flatten* layer is replaced by a *GlobalAveragePooling* layer. The final layer is a *bi-directional LSTM* of size 128.
- Temporal Convolutional Network (TCN) is directly inspired by the Wavenet model [1] which is more convenient for long time series analysis. The specificity of the TCN is to take the full signal as input and use causal dilated convolutions in order to respect the sequence's temporal order. [16] suggest that TCN should be considered as a natural starting point for sequence modeling, machine translation and sentence classification, by empirically proving that it gives a better performance than the recurrent architecture.
  The TCN model is built with 64 filters, a kernel size of 2, a dilation rate from 1 to 16384 (2^{14}) and a dropout rate of 0.2.

Both architectures are trained with the *adam* optimizer with an annealing learning rate from 0.01 to $5e10^{-5}$. We do this by setting a more important decay rate in the optimizer in order to accelerate the decrease. A L2 regularizer on the weights is used with a coefficient of 0.05, knowing that the network can easily over-fit faced to the small dataset.

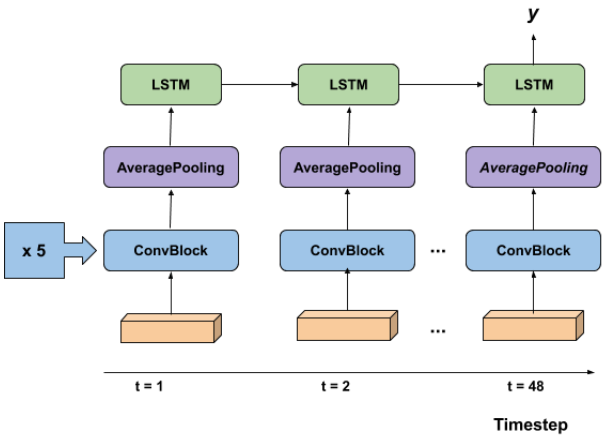

Figure 4. Time-Distributed architecture.

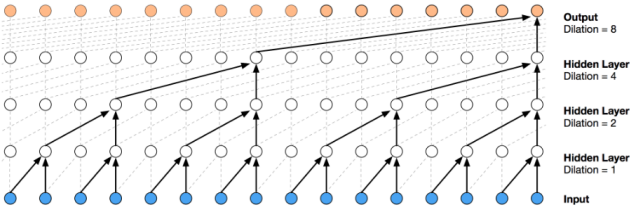

Figure 5. TCN architecture.

4.5 Evaluation metrics

We use two metrics to evaluate the quality of the regression: MAE (mean absolute error) and SMAPE (symmetric mean absolute percentage error) which are commonly used for regression. They have the advantage of being robust to outliers.

$$MAE = \sum_{k=1}^{n} \frac{|\hat{y}_i - y_i^{test}|}{n} \tag{3}$$

$$SMAPE = \frac{2}{n} \sum_{k=1}^{n} \frac{|\hat{y}_i - y_i^{test}|}{|\hat{y}_i| + |y_i^{test}|} \tag{4}$$

4.6 Loss function

We use a classical mean squared error loss function to train the model, defined as follows:

$$Loss = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \tag{5}$$

The loss function $log(cosh)$ have been considered, knowing his good property for managing skewness distribution of target, but doesn't improve results.

5 EXPERIMENTAL RESULTS

5.1 Choice of the best architecture

In order to choose the best model, the architectures presented in Section 4.4 were tested on the whole synthetic dataset (*Baseline Learning* setup defined in Section 4.2. Two variants of the Time-Distributed architecture were also defined: (1) *Time-distributed (seq2point)* (2) *Time-distributed seq2seq* which predicts the full sequence.

Table 1. Baseline regressor results.

Models	MAE (kWh)	SMAPE
Time-distributed	1.76	26.3%
Time-distributed seq2seq	1.82	26.8%
TCN	2.46	39.70%

As can be seen in Table 1, the Time-Distributed architecture obtains the best score both in MAE and SMAPE. We choose this architecture to conduct the experiments described in section 3. The TCN architecture gives promising results but was difficult to train due to the sequence length and the small batch size. We used Horovod library to accelerate the training on GPUs.

Figures 6 and 7 show the MAE and SMAPE measures in relation with the value of water heating consumption in each house. Note that houses 12 and 14 consume excessively compared to other houses but represent only 15% of the data.

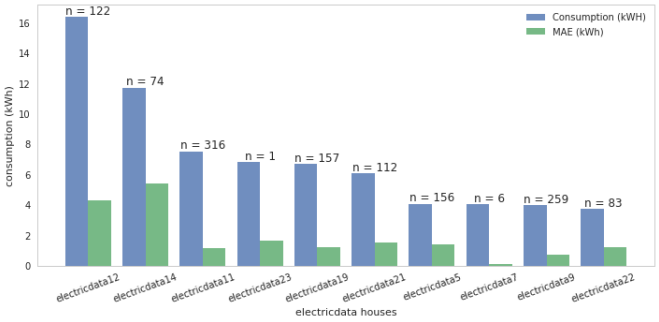

Figure 6. MAE per house for Time-Distributed network.

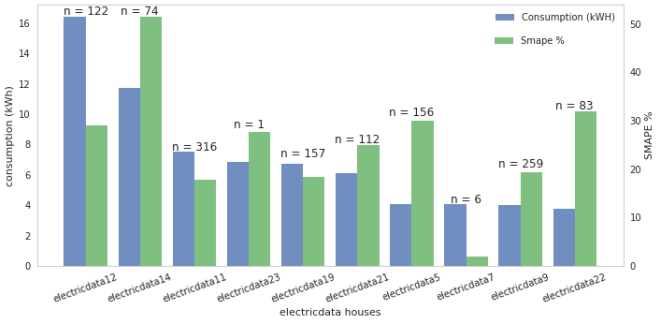

Figure 7. SMAPE per house for Time-Distributed network.

5.2 Cross-validation results

The Time-Distributed architecture which performed the best on the whole synthetic dataset was thus selected for the cross validation procedure. We now want to highlight the fact that a model trained on our synthetic dataset is better than one trained on our real data.

Table 2 presents the results of the procedure for the different training and test datasets, as described in Section 4.2. As for cross-validation, the mean values of MAE and SMAPE were computed over the 10 houses.

Results show that the best performance is obtained by the *Base Learning* and *Full Learning* approaches, i.e. using the synthetic training set. We can notice that transfer learning degrades the performance of the neural network. It can be interpreted by the fact that the network overfits on the real houses despite the low learning rate and that we only train the two last layers. On the other hand, as house profiles are highly diversified, specializing the regressor on a small set of data does not allow to generalize on unseen house.

Table 2. Cross validation results.

Models	Data source	MAE (kWh)	SMAPE
Base learning	MAS	1.92	28.53%
Transfer-learning FL	real	2.55	35.83%
Transfer-learning 2L	real	2.75	37.83%
Full learning	MAS+real	1.92	29.26%
Subset learning	real	3.13	75.71%

6 CONCLUSION

Multi-Agent Simulations (MAS) should be considered as a serious option to solve the NILM problem and build models able to generalize on unseen data. This work shows that a MAS calibrated with real time-use surveys can provide high quality synthetic data and achieve decent performance on disaggregation for high consumption appliances. However, the quality of the generated data and regressor performance could be greatly improved if more real appliance signatures were available. Real annotated data is still important to assess the quality of the prediction. Furthermore, an architecture with a mix of convolutional and recurrent neural network are still a strong baseline choice for sequence modelling, even in the case of long time series prediction.

REFERENCES

[1] Heiga Zen Karen Simonyan Oriol Vinyals Alex Graves Nal Kalchbrenner Andrew Senior Koray Kavukcuoglu. Aaron van den Oord, Sander Dieleman, 'Wavenet: a generative model for raw audio.', *arxiv: arxiv: 1609.03499*, (2016).
[2] Michele Nati Muhammad Ali Imran. Ahmed Zoha, Alexander Gluhak, 'Low-power appliance monitoring using factorial hidden markov models.', *IEEE*, (2013).
[3] Geoffrey E. Hinton. Alex Krizhevsky, Ilya Sutskever, 'Imagenet classification with deep convolutional neural networks.', *Nips*, (2012).
[4] Zongzuo Wang Nigel Goddard Chaoyun Zhang, Mingjun Zhong and Charles Sutton., 'Sequence-to-point learning with neural networks for non-intrusive load monitoring.', *arxiv: 1612.09106*, (2017).
[5] Hart G.W., 'On intrusive appliance load monitoring.', *IEEE*, (1992).
[6] Phil Blunsom. Gábor Melis, Chris Dyer, 'On the state of the art of evaluation in neural language models.', *ICLR*, (2018).
[7] Peng Wang Shibin Qiao Huihui Qiao, Taiyong Wang and Lan Zhang., 'A time-distributed spatio-temporal feature learning method for machine health monitoring with multi-sensor time series.', *Sensors*, (2018).
[8] Matthew J. Johnson J. Zico Kolter, 'Redd: A public data set for energy disaggregation research.', *MIT*, (2011).
[9] William Knottenbelt Jack Kelly, 'Neural nilm: Deep neural networks applied to energy disaggregation', *arxiv: 1507.06594*, (2015).
[10] William Knottenbelt Jack Kelly, 'The uk-dale dataset, domestic appliance-level electricity demand and whole-house demand from five uk homes.', *arxiv: 1404.0284*, (2015).
[11] Bin Yang. Lukas Mauch, 'A new approach for supervised power disaggregation by using a deep recurrent lstm network.', *IEEE*, (2015).
[12] Oliver Parson Haimonti Dutta William Knottenbelt Alex Rogers Amarjeet Singh Mani Srivastava Nipun Batra, Jack Kelly, 'Nilm-tk: An open source toolkit for non-intrusive load monitoring.', *arxiv: 1404.3878*, (2014).
[13] Samuel Cheng. Peng Xiao, 'Neural network for nilm based on operational state change classification.', *arxiv: 1902.02675*, (2019).
[14] Yoshua Bengio. Razvan Pascanu, Tomas Mikolov, 'On the difficulty of training recurrent neural networks.', *arXiv : 1211.5063*, (2012).
[15] Sempé F. Sabouret N. Reynaud Q., Haradji Y., 'Using time use surveys in multi agent based simulations of human activity.', *Proceedings of the 9th International Conference on Agents and Artificial Intelligence - Volume 1: ICAART*, (2017).
[16] Vladlen Koltun Shaojie Bai, J. Zico Kolter, 'An empirical evaluation of generic convolutional and recurrent networks for sequence modeling.', *arxiv: 1803.01271*, (2018).
[17] J.S. Denker D. Henderson R.E. Howard W. Hubbard Y. Le Cun, B. Boser and L.D.Jackel., 'Backpropagation applied to handwritten zip code recognition.', *Neural computation*, (1989).
[18] Men-Shen Tsai. Yu-Hsiu Lin, 'A novel feature extraction method for the development of non-intrusive load monitoring system based on bp-ann.', *arxiv: 1506.04214*, (2015).
[19] M. I. Jordan. Z. Ghahramani, 'Factorial hidden markov models.', *machine learning*, (1997).
[20] Jiawei Lu Jun Xu Gang Xiao Zhipeng Shen, Yuanming Zhang, 'Seriesnet:a generative time series forecasting model.', *2018 International Joint Conference on Neural Networks (IJCNN)*, (2018).