Energy Disaggregation
A Case Study of Large Number of Components
(Machine Learning 2023 Course)

Ilia Kamyshev [1] **Sahar Moghimian** [1] **Bakhtiyar Kazbekov** [1] **Arlan Zhanatbekov** [1] **Arseniy Burov** [1]

Abstract

This paper proposes a neural network architecture to address challenges in energy disaggregation algorithms, which include limited data and the complexity of handling large numbers of individual appliance components. Our proposed model utilizes Independent Component Analysis as a backbone of the network and is evaluated by F1 score for varying numbers of individual appliance working simultaneously. We demonstrate that our model is less prone to overfitting, exhibits low complexity, and effectively decomposes signals with a large number of individual components. We also showed that the proposed model beats existing algorithms on real-world data.

Github repo: arx7ti/ML2023SK-final-project
Presentation file: Presentation on GitHub

1. Introduction

Energy disaggregation is the process of breaking down the total energy consumption of a household into its individual appliance-level components through sophisticated analysis of measurements from a single metering point. This technique has received significant attention in recent years due to its potential to enable greater energy efficiency, demand response, and load forecasting. Energy disaggregation has been extensively studied in the literature, and a variety of techniques have been proposed to perform this task.

The energy disaggregation is also known as Non-Intrusive Load Monitoring (NILM) which scheme is presented in Figure 1. The concept of NILM was first introduced in the 1980s by G. Hart (Hart, 1992), and since then, it has been a popular topic of research in the field of energy management. Various techniques have been suggested to enhance the accuracy of NILM. However, the accuracy can be affected by several factors, such as the total number of appliances, the number of appliances that are working simultaneously, appliances' types, and the measurement noise. Multiple simultaneous appliance switching detection, and correct estimation in practical scenarios with noisy data, remain to be addressed robustly. Recent studies in deep learning-based NILM, such as (Rafiq et al., 2020), have encountered challenges with poor disaggregation accuracy on new unseen households due to a lack of sufficient labeled data, which exacerbates sampling bias and makes models prone to overfitting. Training deep neural networks on limited data may result in higher generalization error and poor disaggregation performance (Rafiq et al., 2021).

So, it is noticeable that the field of energy disaggregation faces several challenges related to algorithm complexity and limited datasets. The typical algorithms are heavy in memory and sensitive to overfitting, which makes them difficult to be ported to the sensor. Moreover, most of the algorithms are trained on a limited number of appliances, while datasets contain dozens of classes of appliances on average. There is also a lack of related studies on the "goodness" of disaggregation, with numerous individual components presented in the aggregated signal. The available datasets have limited combinations of different appliances, biased towards the most frequently used ones, and there is only one dataset available that follows the conventional format, while the rest are raw measurements. These challenges need to be addressed to improve the accuracy and efficiency of energy disaggregation algorithms.

This paper presents a study on the performance of energy disaggregation algorithms for a varying number of individual appliances operating simultaneously. Additionally, we propose a neural network architecture that considers the physics of the energy disaggregation problem by utilizing an unmixing matrix obtained through Independent Component Analysis. We demonstrate that the proposed model is less prone to overfitting, exhibits low complexity, and can effectively decompose signals with a large number of individual components. The algorithm is evaluated by ana-

[1]Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Ilia Kamyshev <ilia.kamyshev@skoltech.ru>.

Final Projects of the Machine Learning 2023 Course, Skoltech, Moscow, Russian Federation, 2023.

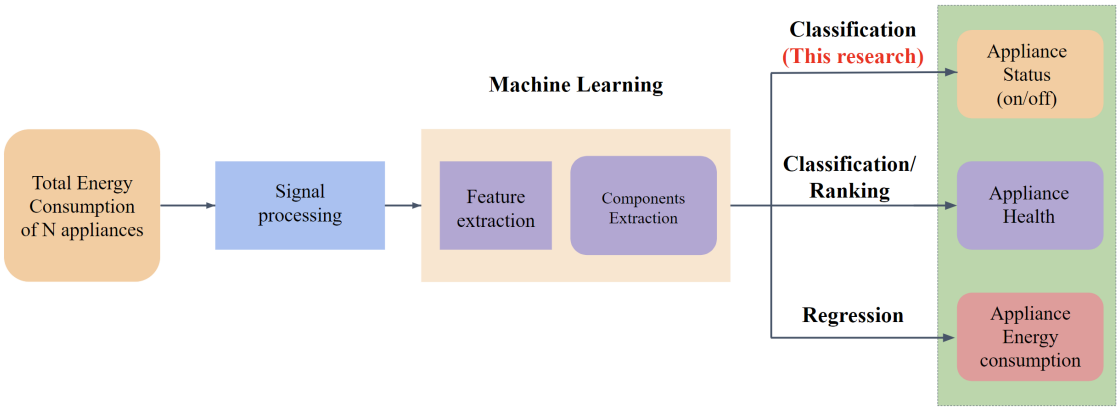

Figure 1. A principle scheme of Energy Disaggregation with the NILM technique based on the work (Gopinath et al., 2020).

lyzing the impact of an increasing number of components on the algorithm's classification accuracy. The results of this study provide the valuable insights for the development of accurate and efficient energy disaggregation algorithms capable of handling complex scenarios and diverse datasets.

2. Related Work

In the 1990s, the NILM techniques were introduced by Hart et al. (Hart, 1992) for monitoring energy consumption at the appliance level using an aggregated signal collected by a single smart meter. Appliance groups were categorized using cluster analysis, and features were extracted and compared to a signature database obtained from individual appliances during training. However, this approach was only tested for steady-state signals and achieved an accuracy of 0.85. Laughman et al. (Laughman et al., 2003), Shaw et al. (Shaw et al., 1998), and Lee et al. (Lee et al., 2005) used power harmonic features obtained from high-sampled signals to differentiate between appliances with similar power consumption. Lee and Braun (Leeb & Kirtley Jr, 1996) used transient power signals with an aggregated power signal and a hierarchical pattern search method to effectively distinguish between appliances. The main advantage of using transient power signals is that they are effective in dividing overlapping events of similar power appliances. However, the sensitivity of transient power signals to electric network topology limits the trained NILM model's applicability to other households.

In recent years, researchers have been drawn to adapt machine learning and deep learning techniques to the NILM domain due to significant advancements in hardware computing power and the improved performance of these techniques in various pattern recognition problems (Gopinath et al., 2020). For instance, in (Cuñado & Linsangan, 2019), the authors utilized the k-NN algorithm with steady-state features to identify individual appliances, achieving good performance. However, the approach was not very effective for detecting unknown appliances.

The proposed approach by (Held et al., 2018) offers a significant advantage as the FIT-PS signal representation fully retains the information of the current signal, including the phase shift. The paper suggests using this representation as a feature for a FeedForward Neural Network, which leads to more robust results compared to standard approaches that use Active Power, Reactive Power, and a fixed number of harmonics. The superiority of the FIT-PS representation is particularly evident in scenarios with numerous simultaneous appliance activations. Additionally, the use of LSTM nets with FIT-PS representation further improves the classification accuracy by incorporating transient state features. However, determining the best initialization for LSTM nets necessitates the use of a validation set due to the problem of multiple local minimums.

The approach based on deep convolutional neural networks using techniques from image segmentation was applied for the NILM problem in paper (Massidda et al., 2020). Our choice of deep convolutional layers is motivated by its automatic feature extraction capabilities as raw load active power signals are given as input. The core part of the proposed architecture is the temporal pooling module that expands the feature dimensionality of input for multilabel classification of multiple active loads operating simultaneously. However, the expansion of feature space in encoder and pooling layers are done at a cost of time resolution reduction of input signal. Although the paper is focused only on classifying only three appliances, the performance of the method needs to be checked for the greater number of components.

3. Datasets

As it was previously mentioned in this project, we are focusing on signals with a high sampling rate to track the electrical properties of an appliance. For this purpose, we selected the PLAID dataset (Gao et al., 2014), which comprises 1800 samples measured at 30 kHz, with a total number of classes equal to 16. Each sample is a pair of voltage and current signals related to one particular class of appliances. The following link can be used for downloading the PLAID data (PLA). The pre-processed data is available in our GitHub repository (Exp).

In addition to the real data, we generated synthetic linearly separable classes of appliances to find a model with the highest disaggregation capability. To generate synthetic classes, we used an algorithm developed by one of the project's authors. The results are not published yet, hence, the code is not freely available yet. The generated and pre-processed data is available in the GitHub repository (Exp).

However, such data cannot be used directly for this task, since the goal of energy disaggregation is to identify those classes of appliances that appear simultaneously in the input signal. For this purpose, we used given sub-metered measurements (signals of individual appliances) and artificially composed them into aggregated signals. This procedure is additionally fair from a physical point of view as long as all the measurements are from the same power grid, i.e., the same voltage level, frequency, etc. Likely, this condition is satisfied. In real circuits, the same process is done within the Kirchhoff law.

A study (Kamyshev et al., 2021) showed that the most frequent number of simultaneously working appliances is equal to eight for a typical household. Since the goal of this research is to investigate the model's performance on numerous components, we generated samples of aggregated signals where from one to n_{classes} types of appliances appear at the same time. Moreover, we also accounted for the fact that some appliances may be duplicated from one to ten times. This is because one particular type of appliance in a household may be presented multiple times, e.g., three phone chargers, two air-conditioners, ten light bulbs, etc. Aggregated samples were generated in a random manner, where each class has an equal probability of being presented in the mixture. By doing so, we aimed to reduce the class imbalance of the resulting dataset.

Priori, signals of the PLAID dataset were re-sampled to 3 kHz. Then we extracted 19,000 regions of interest from the PLAID dataset and generated the same amount of synthetic regions of interest related to standalone appliances. We split them into subsets of shares 70%, 10% and 20% for train, validation and test, respectively. Further, for each subset was used to generate accordingly 7000, 1000, and 2000 of aggregated samples. For each such an aggregated sample corresponds to a binary vector where 1 stands for the class, which is included into the mixture.

4. Algorithms and Models

4.1. Proposed model: ICA+ResNetFFN

For the current project, we proposed the neural network architecture shown on Figure 2.

The incoming aggregated signal X is being decomposed into $n_{\text{classes}} + 1$ components using the un-mixing matrix U, obtained by Independent Component Analysis (FastICA realization), i.e. $X' = XU^T$. Note that the term "+1" stands for a Gaussian component. We used ICA since it underlies the physics of the process in a power grid. One of the requirements of ICA is that the sources are mixed linearly. The aggregated signal is measured at the input node of the power grid, satisfying the Kirchhoff law $i_{agg}(t) = \sum_k i_k(t)$, which is also a linear mixture.

Once X' is obtained, it is linearly projected to a space of dimension d_{model}, i.e. $X_d = X'W^T + b = XU^TW^T + b$. Furthermore, X_d is passed through the sequence of n_{blocks} paired linear layers, followed by ReLU activations and residual connections, as shown in Figure 2. We used $d_{\text{model}} = 64$ and $n_{\text{blocks}} = 15$ in this project. By implementing the relatively simple architecture, we aimed to show that the understanding of the process and nature of the data may guide the selection of an algorithm. It would be fair to note that we are not the first to use ICA as a feature extraction method. On the contrary, there are two works (Giri et al., 2013), (Henriet et al., 2019) that have done it before. However, it is also known that ICA is hard to fit, as it requires many diverse training samples, which is usually a problem in energy disaggregation research. The idea of this project is to show that, on a rich dataset of aggregated signals, ICA is a less complex and more efficient backbone for the classification task.

4.2. Baseline models

4.2.1. Temporal Pooling NILM

The Temporal Pooling NILM (TP-NILM) architecture, adopted from reference (Massidda et al., 2020), is utilized for decomposing the aggregated signal and identifying appliance activation states. This network comprises encoder, temporal pooling, and decoder modules. The encoder module transforms the one-dimensional discrete input signal into a higher-dimensional feature space, consisting of a series of convolutional and max pooling layers with ReLU activation, followed by batch normalization and a dropout layer for regularization. While the encoder reduces the input signal's time resolution, it extracts 256 output features from a single

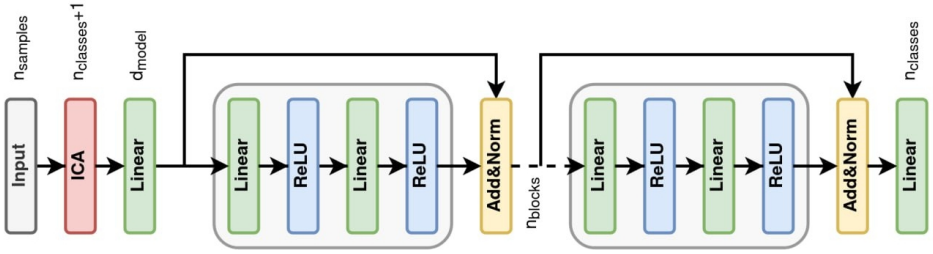

Figure 2. The architecture of the Proposed model so called ICAResNetFFN. The number of parameters for this model is 65,000.

aggregated time-domain signal.

The temporal pooling module accepts the encoder output and generates additional temporal context-aware features using four average pooling layers with different filter settings. These pooling layers further decrease the signal's time resolution while maintaining the feature dimension consistent with the encoder output. The four signals then pass through a convolutional layer with ReLU activation and batch normalization. Additionally, the temporal pooling module performs upsampling to approximate the encoder output's time resolution, merging the detailed features from the encoder module with the additional temporal context features from the temporal pooling module, and passing them to the decoder module. The decoder module reduces the input's feature dimension by passing it through a convolutional layer. Subsequently, the output of the convolutional layer is fed into a fully connected layer that returns an array of scores for appliance-specific activations. Finally, the sigmoid function is applied to obtain values within the range of (0, 1), and appliance activation states are determined using a decision threshold of 0.5.

4.2.2. FIT-PS+LSTM

The FIT-PS method is a novel signal processing method proposed by P.Held and was successfully applied as a feature extraction method for classification in NILM (Held et al., 2018). It consists of three steps. The first step is to divide the sampled signal with respect to the fundamental frequency of the power grid. The division is done by finding the abscissa crossing, namely, the change from negative values of voltage to positive values. Abscissa crossing is chosen because of maximum steepness, almost constant values of derivative for sinusoidal signals, and possible variation of amplitude for other points in the signal. In the second step, using abscissa crossing, the linear approximation of absolute position of zero crossing can be estimated. Finally, indices are assigned to the individual periods by linear interpolation. The resulting matrix $X_{l,k}$ has $n_l \times n_k$ dimensions, where n_l is the number of periods and n_k is the number of sampling points in each period.

The signal passed through the FIT-PS method can be fed to the LTSM network, where n_k is the input dimension. The hidden state dimension is chosen as $1.25 \cdot n_k$. An output of such a network is being averaged across a number of periods and then being passed through a fully connected layer with output size equal to n_{classes}. Finally, the sigmoid layer is used to calculate scores of each class being present in the aggregated signal.

4.2.3. FRYZE+CNN

The CNN model proposed in (Faustine & Pereira, 2020) uses the Fryze power theory (Faustine & Pereira, 2020) and Euclidean distance matrix as feature extraction step for the multi-label classifier. Within the theory, the activation current is decomposed into orthogonal components related to electrical energy in the time-domain:

$$i(t) = i(t)_a + i(t)_f \tag{1}$$

The active current $i(t)_a$ is the current of the resistive load, having the same active power at the same activation voltage. In Fryze's theory, the active power is calculated as the average value of $i(t) \cdot v(t)$ over one fundamental cycle T_s defined as follows

$$i_a(t) = \frac{p_a}{v_{rms}^2} v(t) \tag{2}$$

The active current is therefore defined as

$$i_a(t) = \frac{p_a}{v_{rms}^2} v(t) \tag{3}$$

where the rms voltage v_{rms} is expressed as follows

$$v_{rms} = \sqrt{\frac{1}{T_s} \sum_{t=1}^{T_s} v(t)^2} \tag{4}$$

The current $i(t)_a$ represents the resistance information and is sinusoidal. The non-active component is then equal to

$$i(t)_f = i(t) - i(t)_a \tag{5}$$

Subsequently, $i(t)_a$ and $i(t)f$ undergo two pre-processing steps. Firstly, the signals are dimensionally reduced using Piece-wise Aggregate Approximation. Secondly, the distance matrix for each signal is computed. The two resulting distance matrices are then combined to create input for the multi-label classifier. The classifier is a four-block convolutional neural network, featuring 16, 32, 64 and 128 channels, with kernel sizes of 5×5, 5×5, 3×3, and 3×3 and strides of size 2, respectively. The remaining part of the network comprises three linear layers, consisting of 512, 1024 and n_{classes} neurons, respectively. The main activation function employed is ReLU.

5. Experiments

We conducted two experiments for the synthetic and real classes of appliances, respectively. During each experiment, we trained and validated six models: ICA+ResNetFFN (our proposed model), Fryze+CNN (Faustine & Pereira, 2020), Temporal Pooling NILM (Massidda et al., 2020), FIT-PS+LSTM (Held et al., 2018), ICA+k-NN, and ICA+Random Forest. The last two models were chosen as they showed significant performance on the ICA features (). Each experiment required 45 minutes of processing time on a machine with 2× RTX 2080 Ti GPUs and 128GB of RAM.

We assessed the performance of classification algorithms using the F_1-score, averaged over samples. To get a complete picture of the disaggregation performance, we computed F_1-score for each number of appliances present simultaneously, that it for the range from 1 to n_{classes} and for the whole sample. Ideally, the distribution of F_1-score across a different number of simultaneously working appliances should be uniform. That is, there should not be a bias towards a particular number of appliances at a runtime. Intuitively, with the number of appliances growing F_1-score is expected to drop. This is due to the fact that some appliances may have dozens (30-50) harmonics, which can be treated as noise while appearing together with dominant appliances which have the first from five to ten strong harmonics.

6. Results

6.1. Experiment 1: Synthetic Appliances

Figures 3 and 4 present the outcomes of training and validating four deep learning models using synthetic data. Our proposed model demonstrates the lowest validation loss and highest F_1-score, exhibiting smoother convergence. Furthermore, Figure 5 reveals that our model sustains a consistent F_1-score across varying numbers of concurrently operating appliances, whereas the other models exhibit a decline between two and ten individual components.

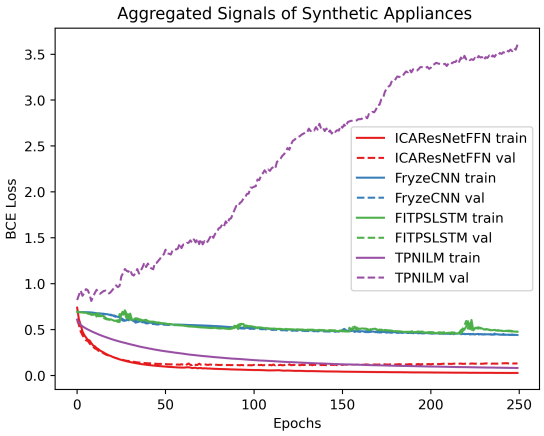

Figure 3. Binary Cross Entropy loss for 4 deep learning models.

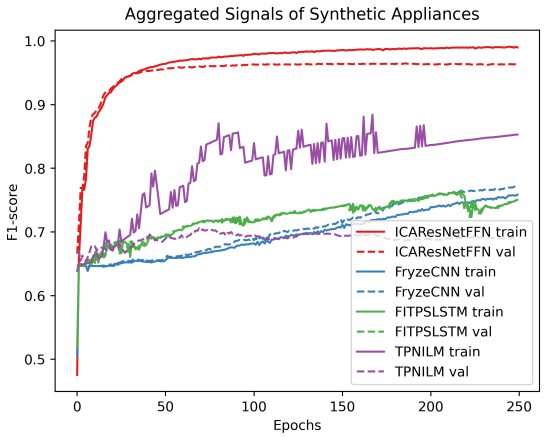

Figure 4. F_1-score (sample average) for four deep learning models.

6.2. Experiment 2: Real Appliances

Figures 6 and 7 display the binary cross-entropy loss and F_1-score for training and validation of four deep learning models on real data, respectively. Our proposed model achieves the lowest validation loss and highest F_1-score, exhibiting smoother convergence than the other models. However, Figure 8 reveals that our model no longer sustains a uniform F_1-score across varying numbers of concurrently operating appliances. Nonetheless, it demonstrates a higher F_1-score where the other algorithms experience performance drops. The results are summarized in Table 1, which compares the F_1-scores (sample averaging) of all models.

The t-SNE embeddings of real and synthetic appliance classes can elucidate the performance differences observed in experiments. Figure 9(a) reveals the complex structure of real appliance classes, with some classes exhibiting multiple data-point clusters or overlapping with others. This

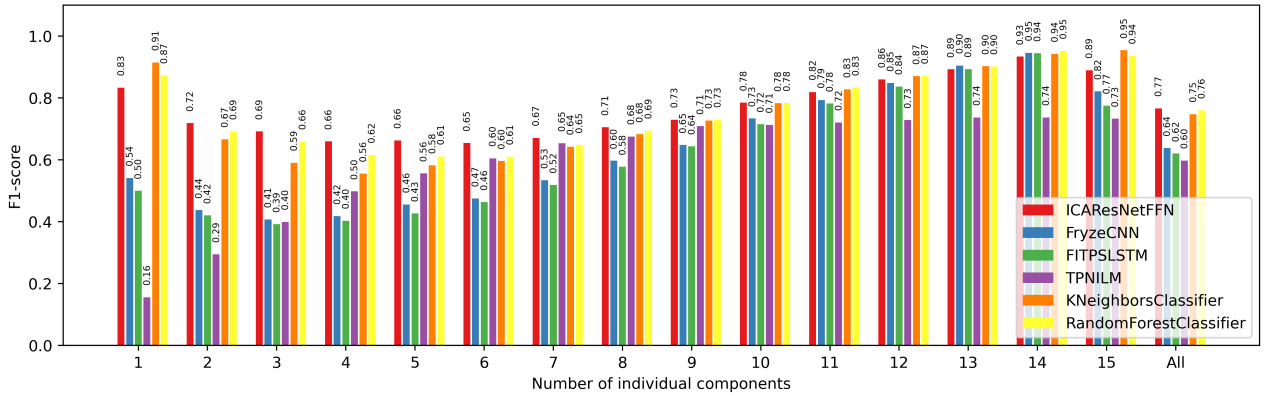

Figure 5. Distribution of F_1-scores (samples averaging) across different number of simultaneously working appliances for four deep learning models and 2 classical machine learning models.

Table 1. Average accuracy for all models

Experiment	*ICA+ResNetFFN*	*Fryze+CNN*	*FIT-PS+LSTM*	*Temporal Pooling NILM*	*k-NN*	*RandomForest*
Real data	0.77	0.64	0.62	0.6	0.75	0.76
Synthetic data	0.95	0.68	0.72	0.67	0.88	0.93

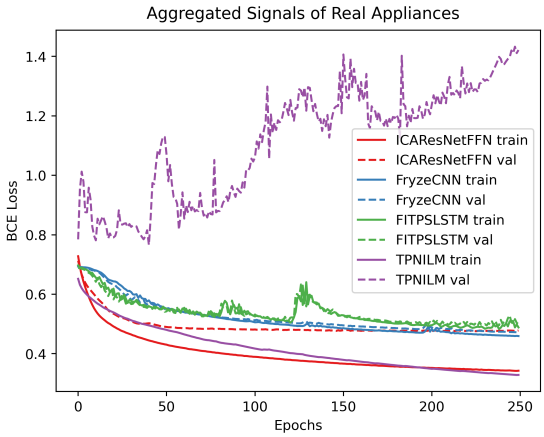

Figure 6. Binary Cross Entropy loss for four deep learning models.

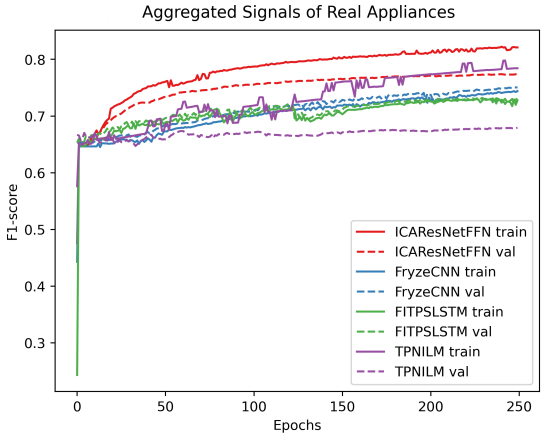

Figure 7. F_1-score (sample average) for four deep learning models.

overlap occurs naturally due to appliances containing common electrical elements. For instance, washing machines have heating elements, motors, and water pumps, while heating elements are also the primary component of water kettles. Consequently, when operating simultaneously, these appliances may be misidentified as a single device, leading to reduced performance. Developing a disaggregation algorithm solely on real data may not accurately capture its generalization capability. Employing synthetic data with linearly separable classes, as shown in Figure 9(b), can guide the development of an optimal architecture that can be subsequently applied to real data.

7. Conclusion

In conclusion, our research has demonstrated the importance of considering the underlying physics of the data when selecting an appropriate model to accurately capture the physical characteristics of the energy disaggregation problem. We employed Independent Component Analysis (ICA) as our method of choice, as it inherently assumes that the signals are linearly mixed, which aligns with Kirchhoff's circuit laws.

A primary objective of this research was to evaluate the performance of the algorithm in scenarios with varying numbers of concurrent appliances. To ensure that our dataset

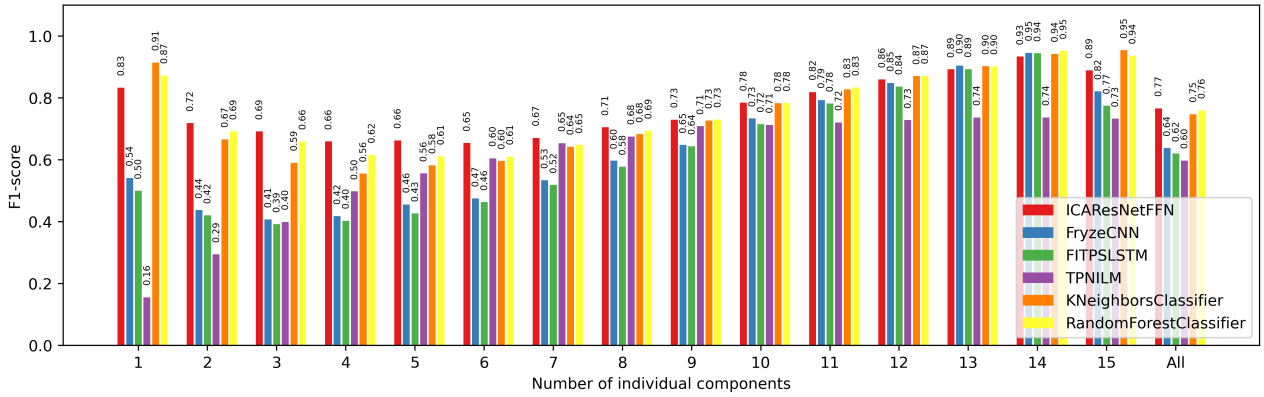

Figure 8. Distribution of F_1-score (samples averaging) across different number of simultaneously working appliances for 4 deep learning models and 2 classical machine learning models.

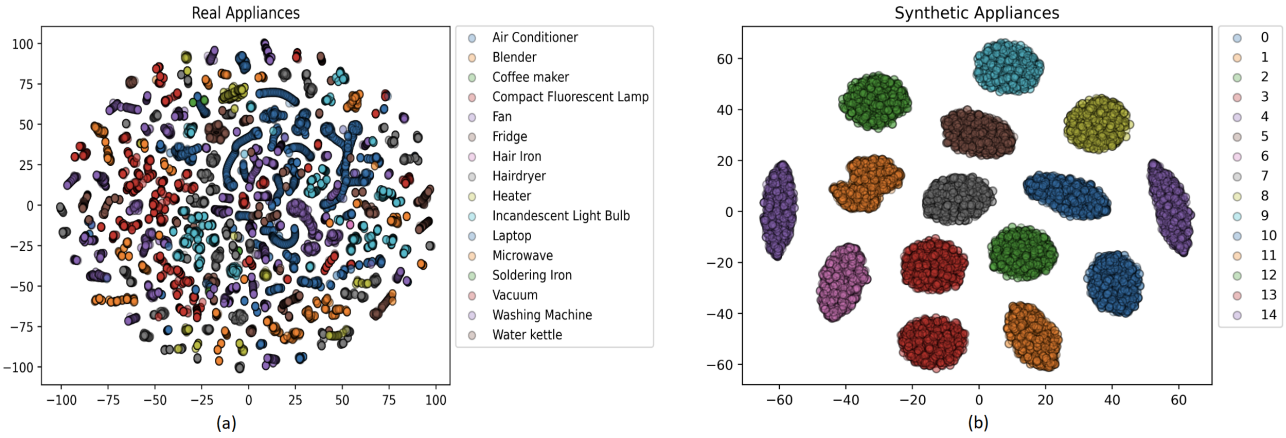

Figure 9. t-SNE visualization of the feature space for real data: (left) Real data, (right) Synthetic data.

was sufficiently diverse to accommodate all possible combinations of components, we deliberately curated a rich and comprehensive dataset.

Our findings indicate that ICA exhibits exceptional performance when applied to large datasets with abundant training examples for any number of simultaneous appliances. Our model has the best performance over modern, existing baseline models. In addition, the suggested model better tackles the problem of data imbalance, which was not properly solved for yet. This study contributes valuable insights into the effective application of ICA for modeling and analyzing complex physical systems with multiple interacting components.

References

Experiment data. https://github.com/arx7ti/ML2023SK-final-project. Accessed: 2023-03-26.

Plug load appliance identification dataset (plaid) data. https://figshare.com/articles/dataset/PLAID_-_A_Voltage_and_Current_Measurement_Dataset_for_Plug_Load_Appliance_Identification_in_Households/10084619. Accessed: 2023-03-26.

Cuñado, J. and Linsangan, N. A supervised learning approach to appliance classification based on power consumption traces analysis. In *IOP Conference Series: Materials Science and Engineering*, volume 517, pp. 012011. IOP Publishing, 2019.

Faustine, A. and Pereira, L. Multi-label learning for appliance recognition in nilm using fryze-current decomposition and convolutional neural network. *Energies*, 13(16): 4154, 2020.

Gao, J., Giri, S., Kara, E. C., and Bergés, M. Plaid: a public dataset of high-resoultion electrical appliance measurements for load identification research: demo abstract.

In *proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, pp. 198–199, 2014.

Giri, S., Bergés, M., and Rowe, A. Towards automated appliance recognition using an emf sensor in nilm platforms. *Advanced Engineering Informatics*, 27(4):477–485, 2013. ISSN 1474-0346. doi: https://doi.org/10.1016/j.aei.2013.03.004.

Gopinath, R., Kumar, M., Joshua, C. P. C., and Srinivas, K. Energy management using non-intrusive load monitoring techniques–state-of-the-art and future research directions. *Sustainable Cities and Society*, 62:102411, 2020.

Hart, G. W. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, 1992.

Held, P., Mauch, S., Saleh, A., Abdeslam, D. O., and Benyoucef, D. Frequency invariant transformation of periodic signals (fit-ps) for classification in nilm. *IEEE Transactions on Smart Grid*, 10(5):5556–5563, 2018.

Henriet, S., Şimşekli, U., Santos, S. D., Fuentes, B., and Richard, G. Independent-variation matrix factorization with application to energy disaggregation. *IEEE Signal Processing Letters*, 26(11):1643–1647, 2019. doi: 10.1109/LSP.2019.2941428.

Kamyshev, I., Kriukov, D., and Gryazina, E. Cold: Concurrent loads disaggregator for non-intrusive load monitoring, 2021.

Laughman, C., Lee, K., Cox, R., Shaw, S., Leeb, S., Norford, L., and Armstrong, P. Power signature analysis. *IEEE power and energy magazine*, 1(2):56–63, 2003.

Lee, K. D., Leeb, S. B., Norford, L. K., Armstrong, P. R., Holloway, J., and Shaw, S. R. Estimation of variable-speed-drive power consumption from harmonic content. *IEEE Transactions on Energy Conversion*, 20(3):566–574, 2005.

Leeb, S. B. and Kirtley Jr, J. L. Transient event detector for use in nonintrusive load monitoring systems, January 9 1996. US Patent 5,483,153.

Massidda, L., Marrocu, M., and Manca, S. Non-intrusive load disaggregation by convolutional neural network and multilabel classification. *Applied Sciences*, 10(4):1454, 2020.

Rafiq, H., Shi, X., Zhang, H., Li, H., and Ochani, M. K. A deep recurrent neural network for non-intrusive load monitoring based on multi-feature input space and post-processing. *Energies*, 13(9), 2020. ISSN 1996-1073. doi: 10.3390/en13092195.

Rafiq, H., Shi, X., Zhang, H., Li, H., Ochani, M. K., and Shah, A. A. Generalizability improvement of deep learning-based non-intrusive load monitoring system using data augmentation. *IEEE Transactions on Smart Grid*, 12:3265–3277, 2021.

Shaw, S. R., Abler, C., Lepard, R., Luo, D., Leeb, S. B., and Norford, L. K. Instrumentation for high performance nonintrusive electrical load monitoring. *Journal of Solar Energy Engineering*, 1998.

A. Team member's contributions

Explicitly stated contributions of each team member to the final project.

Ilia Kamyshev (20% of work)

- Proposition of this study's idea
- Implementation the data generator and ICA+ResNetFFN algorithm
- Performing experiments and plotting the results

Sahar Moghimian (20% of work)

- Literature review on ICA in energy disaggregation
- Writing introduction, results, and conclusion
- Reviewing the whole paper
- Making presentation

Bakhtiyar Kazbekov (20% of work)

- Writing 50% of related work section
- Describing Temporal Pooling NILM model in the paper
- Implementing 50% of Temporal Pooling NILM model

Arlan Zhanatbekov (20% of work)

- Writing 50% of related work section
- Describing FIT-PS
- Implementing 50% of Temporal Pooling NILM model

Arseniy Burov (20% of work)

- Training classical machine learning models on ICA features
- Writing about Fryze Current Decomposition
- Writing about PLAID dataset
- Pre-processing the PLAID dataset
- Preparing the GitHub Repo

B. Reproducibility checklist

Answer the questions of following reproducibility checklist. If necessary, you may leave a comment.

1. A ready code was used in this project, e.g. for replication project the code from the corresponding paperformance was used.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   General comment: If the answer is **yes**, students must explicitly clarify to which extent (e.g. which percentage of your code did you write on your own?) and which code was used.

   Students' comment: All the code was written by our team. The architectures and hyper-parameters of the baseline DL models were inherited, but their practical implementation was also thoroughly done by us.

2. A clear description of the mathematical setting, algorithm, and/or model is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

3. A link to a downloadable source code, with specification of all dependencies, including external libraries is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

4. A complete description of the data collection process, including sample size, is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

5. A link to a downloadable version of the dataset or simulation environment is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: The real data and pre-processed synthetic data are available on the GitHub repository. However, the raw data and the code to generate it are unavailable, as it is the unpublished material.

6. An explanation of any data that were excluded, description of any pre-processing step are included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

7. An explanation of how samples were allocated for training, validation and testing is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

8. The range of hyper-parameters considered, method to select the best hyper-parameter configuration, and specification of all hyper-parameters used to generate results are included in the report.

   ☐ Yes.
   ☐ No.
   ☑ Not applicable.

   Students' comment: Our model is based on CNN, where we selected reasonable and more or less standard hyper-parameters. We did not observe overfitting and other problems, hence, we did not further tune the model as it would take too much time. DL baseline models had all the necessary hyper-parameters described in their articles. The tuning of classical ML models was done via Grid Search. However, they are not the main research interest, therefore, the parameters are not specified in the report and can be found in the provided and described code.

9. The exact number of evaluation runs is included.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

10. A description of how experiments have been conducted is included.

    ☑ Yes.
    ☐ No.
    ☐ Not applicable.

Students' comment: None

11. A clear definition of the specific measure or statistics used to report results is included in the report.

    - [x] Yes.
    - [ ] No.
    - [ ] Not applicable.

    Students' comment: None

12. Clearly defined error bars are included in the report.

    - [x] Yes.
    - [ ] No.
    - [ ] Not applicable.

    Students' comment: None

13. A description of the computing infrastructure used is included in the report.

    - [x] Yes.
    - [ ] No.
    - [ ] Not applicable.

    Students' comment: None