Energy disaggregation and appliance detection in the context of NILM using Deep Learning

Victor Guizien Martin

Author

Eugenio Sánchez Ubeda

Instituto de Investigación Tecnológica (IIT) – Comillas

Director

Andrea Veiga Santiago

Endesa

Director

***Abstract*—Research in the field of NILM (*non-intrusive load monitoring)* has seen an unprecedented surge in the past years mainly due to the increase in domestic household energy monitoring, finally making this alternative viable. Over these years, researchers have tried and implemented multiple techniques and algorithms that managed, with a certain degree of success, to perform energy disaggregation. However, these techniques are often very niche, do not generalize very well because of their complexity, or they simply do not perform all that great. This paper proposes the vizgram, a new approach on how to process and interpret energy data that serves as the model input, transforming it into an image that can later be fed to a Convolutional Neural Network, taking advantage of the already proven usefulness and remarkable performance of these types of architectures in image classification. The proposed approach has been applied with success to a real case consisting of five different appliances.**

Keywords—NILM, energy disaggregation, MLP, CNN, DNN.

I. INTRODUCTION

As energy consumption continues to grow exponentially across the world, so does the need of monitoring, managing, and reducing its usage as much as possible, particularly in domestic households. In the European Union alone, the energy consumption of residential buildings reached 30% of the total energy consumption [1]. Moreover, the continued growth in this field is causing a plethora of problems, such as depleting energy sources or increasing greenhouse gas emissions due to the energy production process, shining light on the importance of reducing said consumption by conserving energy and modernizing the electrical grid infrastructure.

In order to achieve this modernization, one key element is energy monitoring. The energy end-use monitoring has received a widespread attention over the last two decades, either in a non-intrusive fashion (NILM) using smart meters which aim to reduce the cost of installation and maintenance [2, 3] or distributed sensors and smart outlets that became available through recent technological developments [4, 5].

Non-intrusive load monitoring (NILM) aims to break down a households' aggregate electricity consumption into individual appliances, achieving three different objectives. Firstly, it informs households' occupants of how much energy each appliance consumes, which has proven to raise consumers' awareness and prompt them to take proactive steps in reducing their own consumption [6, 7]. Secondly, the detailed feedback of energy consumption provided by the system can result in economic savings if, for instance, it recommends replacing inefficient appliances with new and more efficient ones. Lastly, the NILM system could also be able to give advice or even automate appliances to defer usage to given timeframes where electricity is cheaper.

NILM is achieved by installing a smart meter connected to the mainline, which records electrical energy consumption at a given interval and communicates with a central server where the information is stored. Compared to its counterpart, ILM (intrusive load monitoring), this alternative is far cheaper and easier to deploy and scale, which is why it is currently being massively installed across the EU where the objective by 2020 was to achieve an installation rate of 80% of all the electrical measurement points [8].

The NILM system mainly consists of three steps: (1) data acquisition, (2) feature extraction and (3) data learning and identification. A rich dataset is needed for (1), while Machine Learning algorithms are required for (2) and (3). In order to train a NILM based model, the dataset has to contain appliance-level energy consumption with a certain reading frequency, which dictates the methodology to tackle this problem. For instance, high frequency appliance-level readings are used to identify appliances' load signatures and requires specific and expensive hardware [9], while lower frequencies (1Hz and lower) are often used to look for more traditional data patterns and can be polled with more traditional smart meters.

Depending on the scope of the project and the resources available, NILM can aim to solve a regression problem or a classification one. For this project, due to the relatively short timeframe and limited resources, it was decided that this project would be a classification one. That is, our developed model is trained to recognized whether an appliance is on or off at any given time, whilst only having the total energy as its input.

In the past, researchers have shown variable results while trying to solve NILM as a regression problem [10, 11] but rarely do they focus on the simpler task of identifying which appliance was activated at a given time, information that is rich enough in itself and is able to provide plenty of insight on consumption habits.

The remainder of the paper is structured as follows. In Section II we describe the proposed innovative way of data preparation for our model input, while construction of the model output is described in Section III. We then describe in Section IV the models used, how and what parameters were chosen, and the different difficulties encountered in this process. Finally, conclusions are drawn in Section V.

II. VIZGRAMS

As explained previously, this paper aims to use CNNs models to try and solve a classification problem in the field of NILM. In order to introduce the reader to the concept of vizgrams, it is first necessary to briefly explain why they are necessary and how they were created.

In a NILM problem, the traditional input is the total energy consumption of the household, recorded at a given interval of time. For our model to learn and properly disaggregate this signal of the total energy consumption, we also need the detail of what appliances were consuming.

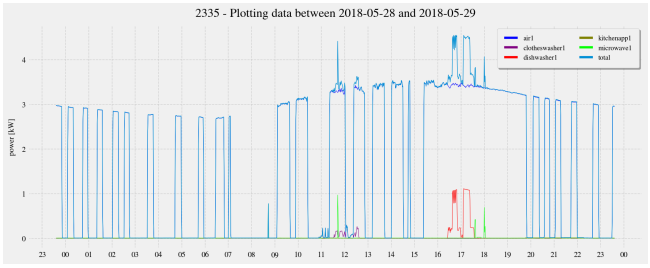

Figure 1 – Example of data

Figure 1 shows an example of data where we can see both the total signal and the consumption of individual appliances. The dataset used for this paper has been the Dataport Dataset from Pecan Street [12], mostly due to its greater number of sampled appliances (20+ per house), number of houses (50 houses worth of data for free for students and researchers), adequate sampling rate for this problem (15min, 1min and 1sec disaggregate and aggregate data) and a long enough period of recorded values (1 year). This dataset offered recordings of 50 houses, 25 in Austin and 25 in New York. The recorded appliances slightly differed from house to house but overall remained the same. After an initial dataset exploration, we decided to work with a single house of Austin and 5 main appliances: air1, dishwasher1, clotheswasher1, microwave1 and kitchenapp1 (kitchen circuit where smaller appliances like toaster or blender are connected). The minutely data of a single house was already enough for our work environment to struggle with data processing, and the appliances were chosen based on what was subjectively considered as most important in a day-to-day basis.

As said previously, the input data for NILM models has traditionally been the raw data of the total energy consumption. Instead of making use of this standard way of looking at the data, we constructed a matrix that compiles this very information, as well as the energy consumption *before* and *after* the instant that is being observed. In order to construct this matrix, several parameters need to be defined first.

A. Window

To add the energy information of the past and future of a certain instant t_0, we need to define how much *before* and *after* we want to look for. Initially, this parameter was set to 12h (12h before and 12h after t_0), resulting in a window length of 24h. This parameter was not altered during the rest of the project.

B. Sampling rate

This is the rate at which the signal of the total consumption will be sampled at *within* the defined window. We also need to define how many sampling rates we would like. For instance, choosing a sampling rate of 1 minute means that the signal will be sampled at this rate before and after the instant t_0, until the length of the previously defined window is reached. In this example, we would have $12h \cdot 60m/1 = 720$ samples before and after t_0.

As we want to construct a matrix, the number of samples has to be the same for each sampling rate. This means that the maximum samples will be defined by the largest sampling rate that we decided upon, and that smaller sampling rates will not reach the maximum window length. For example, if we choose 4 sampling rates, 1, 2, 5 and 10 minutes. The maximum samples will be defined by the following equation.

$$\frac{24 \cdot 60}{10} = 144 \quad (1)$$

And, if we calculate the maximum number of minutes that the different polling rates will be able to reach:

- 1 minute: $1 \cdot 144 = 144\,min$, which is 2h24, or 1h12 before and after t_0
- 2 minutes: $2 \cdot 144 = 288\,min$, which is 4h48, or 2h24 before and after t_0
- 5 minutes: $5 \cdot 144 = 720\,min$, which is 12h, or 6h before and after t_0
- 10 minutes: $10 \cdot 144 = 1440\,min$, which is 24h, or 12h before and after t_0

This means that the more detail we get about the signal (smaller sampling rates), the less window will be covered. Inversely, the less detail we get about the signal (greater sampling rates), the more window will be covered, until reaching its maximum length.

Figure 2 illustrates the described process using different sampling rates.

Figure 2 – Example of sampling data with various sampling rates

The chosen sampling rates for the rest of the work are 1 to 10 minutes, with 1-minute increments.

C. Stride

The last parameter that needs to be determined for the vizgram is the *stride*. This is simply the time interval between each computed vizgram. This will determine the number of total vizgrams computed. For instance, if we have 353 days of data (the first and last day of data have discarded in order to compute the vizgrams at the hour 00 of the day 2 and the day 364, as the vizgram overflows by 12h), and we are using a window length of 24h, using a stride of 3h will produce:

$$\frac{363 \cdot 24}{stride} = \frac{363 \cdot 24}{3} = 2904\ vizgrams \quad (2)$$

The matrix result of a vizgram has the following structure:

$$\begin{matrix} t_{0-n_{samp_1}} & t_{0-(n-1)_{samp_1}} & \cdots & t_{0_{samp_1}} & \cdots & t_{0+(n-1)_{samp_1}} & t_{0+n_{samp_1}} \\ t_{0-n_{samp_2}} & \cdots & & & & & \vdots \\ \vdots & & & & & & \\ t_{0-n_{samp_i}} & t_{0-(n-1)_{samp_i}} & \cdots & t_{0_{samp_i}} & \cdots & t_{0+(n-1)_{samp_i}} & t_{0+n_{samp_i}} \end{matrix}$$

where n is the number of samples determined by (1) divided by 2.

So, for example, $t_{0-n_{samp1}}$ would be t_{0-72_1}, which means the instant t_0 minus 72 samples of 1 min, meaning the total consumption 72 minutes before t_0.

Additionally, the top 3 rows of the vizgram contain information relative to the time where it was sampled. For example, if the vizgram is computed at the hour 6 on a Wednesday in August, the top 3 would be as depicted in Figure 3, were we simply constructed an array of '1' in the corresponding zone, proportional to the data in question.

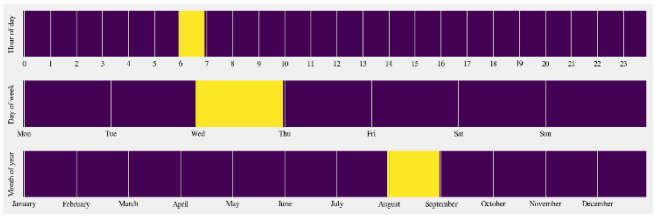

Figure 3 – Example of temporal features of vizgram

With all this information, we are finally ready to construct a whole vizgram, as shown in Figure 4:

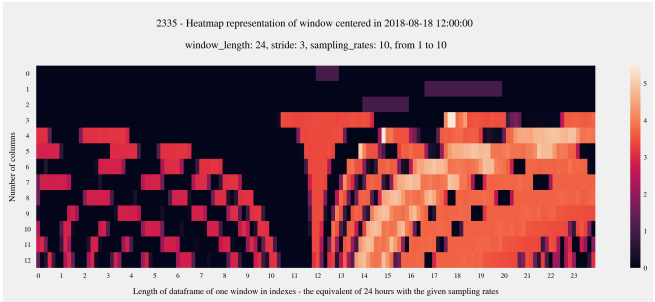

Figure 4 – Vizgram

To better understand what the vizgram is representing, the next figures highlight the equivalence between the vizgram and the data of the total consumption for the first two rows of said vizgram (excluding the top 3 which contain information relative to the date).

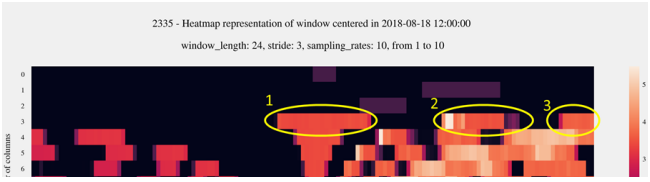

Figure 5 – Identification of power peaks in vizgram – first row

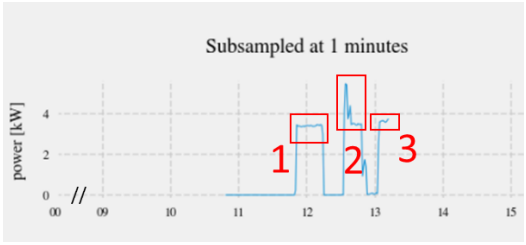

Figure 6 – Identification of power peaks in signal representation – first row

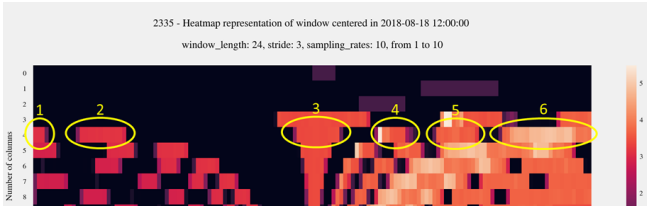

Figure 7 – Identification of power peaks in vizgram – second row

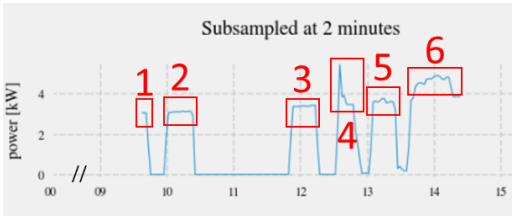

Figure 8– Identification of power peaks in signal representation – second row

Figure 5 and Figure 6 represent the smallest sampling rate (1 min). As the largest sampling rate chosen was 10 minutes, the number of samples is the same as (1), which was 144. That means that our first row of the vizgram represents the total consumption 72 minutes prior to t_0 and 72 after t_0, each minute. The brighter, more orange looking colors of the vizgram correspond to the higher consumption values of the original data (see the color bar of Figure 5).

If we focus on Figure 7 and Figure 8, the sampling rate is now 2 minutes, meaning that this row will extend 144 minutes before t_0 and 144 minutes after t_0. Extrapolating this analysis to the rows below, it becomes visible why the vizgram has this umbrella/fountain-like shape. As we use higher sampling rates but are restricted to a limited number of samples, we reach further back and ahead from t_0, meaning that each sampling rate will sample what the previous one already did, but at a lesser level of detail.

Figure 9 illustrates this idea in the vizgram, while Figure 10 shows the equivalent peak in the traditional time series representation with a red arrow (in this picture, hours 15 to 24 have been cut for better representation purposes).

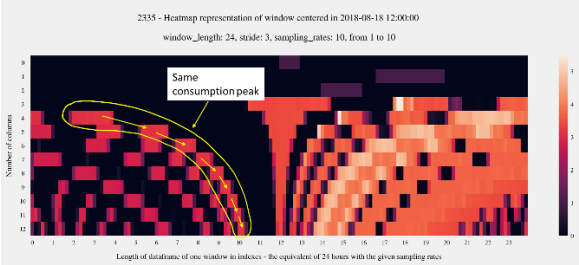

Figure 9 – Identification of power peaks across different sampling rates in vizgram

Figure 10 – Identification of power peak across different sampling rates in the signal plot

The values of the different parameters required to construct the vizgram had to be modified throughout the project due to various reasons explained in the following section.

III. Output and Data Processing

A. Output

With our input data ready for machine learning models, we only need to construct and pre-process the output data.

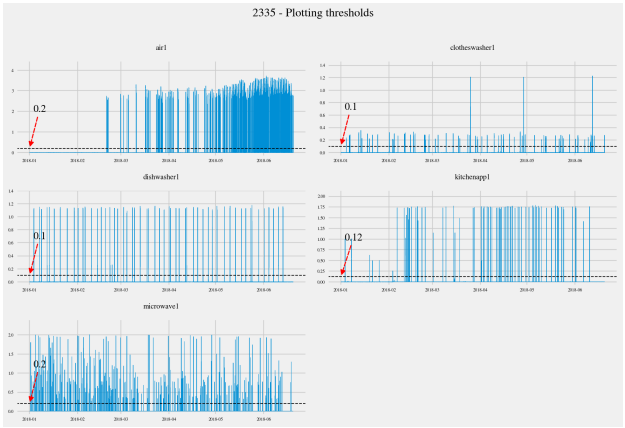

Figure 11 – Initial thresholds of all appliances

As explained earlier, this problem will try to determine which appliance was ON or OFF at the sampled time. In order to do this, we need to define what is considered ON or OFF based on a previously defined power threshold. When the signal of the appliance is above this threshold, it will be considered ON, and when below, OFF. These thresholds were initially set by visual inspection of the temporal data of all appliances, as depicted in Figure 11.

B. Data processing

With input and output data parameters defined, we created a set of vizgrams for each measured instant. As our dataset contains a year of data and the first stride was set to 3h, that means that we obtain 2904 vizgrams (2).

Figure 12 shows how we are detecting the different appliances based on our defined activation thresholds and at the defined stride. As we can see, this stride is clearly too high, and we are missing a lot of information. Based on this realization, the stride was set to 1 min (the lowest possible value given our dataset has data every minute).

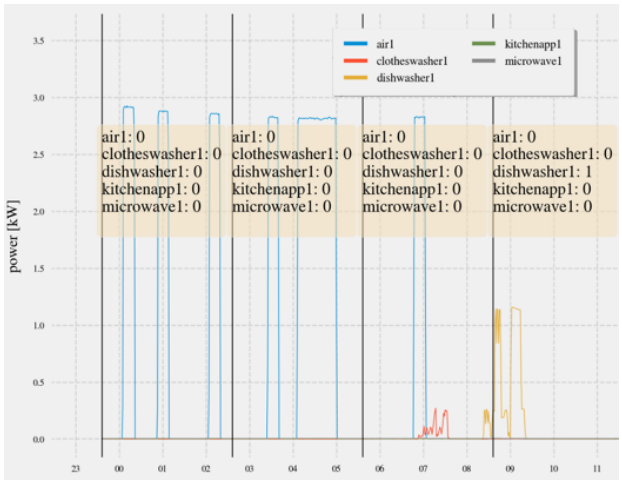

Figure 12 – Detected activations based on thresholds (stride 3h)

With a stride of 1 minute, we have:

$$\frac{363 \cdot 24}{stride} = \frac{363 \cdot 24 \cdot 60}{1} = 522\,720\ vizgrams \tag{3}$$

We can see in Figure 13 the detected activations now seem more reasonable.

Figure 13 – Detected activations based on thresholds (stride 1min)

C. *Output representation*

A heatmap representation of the output signal for each appliance was created to have a general visualization of how a particular appliance is being used and to check how the defined thresholds were performing.

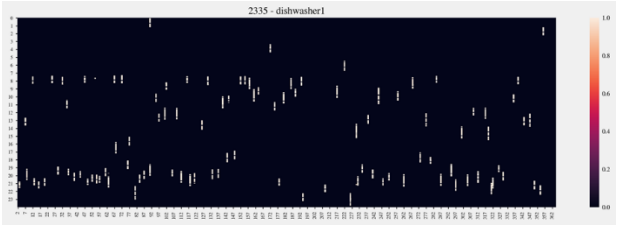

Figure 14 – Representation of output Y – dishwasher

Figure 14 shows this representation for the dishwasher, where rows represent the hour of the day and each column is a day of the dataset. In this figure, we can clearly see that the threshold set for this appliance was perhaps a bit too high, as most activations seem to be cut in half (one would expect a person to set the dishwasher once a day, two at most perhaps, but certainly not three consecutive times).

To solve this, we first lowered the threshold and obtained the result shown in Figure 15:

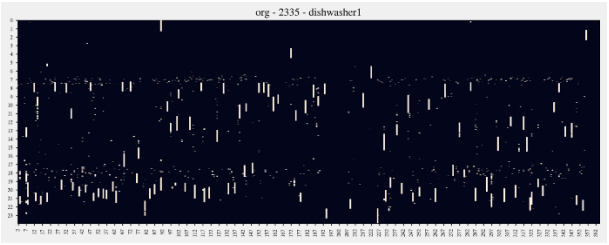

Figure 15 – Representation of output Y with redefined threshold – dishwasher

In Figure 15 we can see that now activations seem more realistic, but that we also captured a lot of noise.

This noise can be filtered by designing a relatively simple filtering algorithm that iterates over the whole matrix and stops at every '1' it finds, checking the number of '1' backwards and forwards within a window of size w_size. Subsequently, the algorithm compares these two results to a backward and forward threshold, respectively. If both results are lower than their corresponding threshold, the evaluated '1' is considered noise and replaced with a '0'. Figure 16 shows the result of applying the algorithm to the dishwasher of Figure 15 with these parameters:

- w_size: 60
- $backward\_threshold$: 7
- $forward\_threshold$: 50

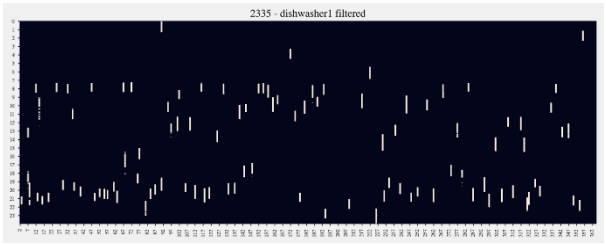

Figure 16 – Y data with filter algorithm – dishwasher

However, we can still see that some activations are not complete. To solve this, a similar algorithm was designed to fill the gaps. Its behavior is analogous to the previously explained algorithm, but instead of stopping at '1', it does at '0'. Figure 17 shows the result of applying the filling algorithm to the dishwasher with these parameters:

- w_size: 50
- $backward\_threshold$: 40
- $forward\_threshold$: 10

Both algorithms were applied for the dishwasher and clotheswasher, and definitive values used to construct the output variables will be summarized in section V.

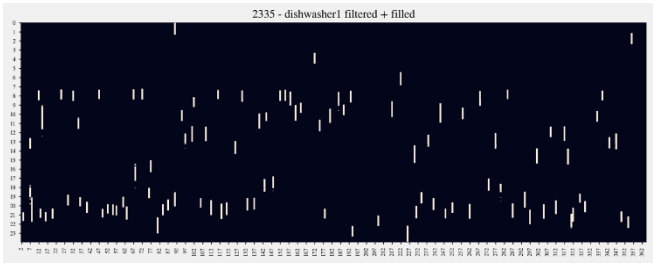

Figure 17 – Y data with filter and fill algorithm – dishwasher

Now that the input and output data has been defined and constructed, we are ready to use them in our machine learning models.

IV. Case Study

Two models were created to evaluate our data (which was, as explained earlier, from a single house with ID 8336), a Multilayer Perceptron (MLP) and a Convolutional Neural Network (CNN). The former was needed as a reference model to assess if the CNN was able to offer better results, as the idea behind the vizgrams was to use them in CNNs in the first place. Because the vizgram is simply a visual representation of a matrix, this data can be used and fed as input to an MLP without further data processing, establishing an easy baseline to which compare the results of the CNN.

For the MLP, we created one model for each appliance, consisting of a single hidden layer with 15 neurons. As for the CNN, we created one model for all the appliances to be able to evaluate a greater number of structures. We first focused on smaller, less complex structures consisting of a single convolutional layer with no regularization, and steadily incremented the complexity of the model.

It is important to note that we are dealing with a heavily imbalanced problem, where the proportion of '0' is far greater than that of '1', as we can see in Figure 18. For example, the dishwasher is only used 3.146% of the total minutes of the year.

```
Number of 0 air1: 475676 - prop: 91.0%
Number of 1 air1: 47044 - prop: 9.0%

Number of 0 clotheswasher1: 515820 - prop: 98.68%
Number of 1 clotheswasher1: 6900 - prop: 1.32%

Number of 0 dishwasher1: 506273 - prop: 96.854%
Number of 1 dishwasher1: 16447 - prop: 3.146%

Number of 0 kitchenapp1: 520875 - prop: 99.647%
Number of 1 kitchenapp1: 1845 - prop: 0.353%

Number of 0 microwave1: 520967 - prop: 99.665%
Number of 1 microwave1: 1753 - prop: 0.335%
```

Figure 18 – On/Off proportion of data

This means that our models must be greater that proportion, as otherwise it would be more efficient to simply decide based upon the most frequent class, also known as the no information rate.

Table 1 sums up the results of our reference MLP model and the best CNN we were able to design. Furthermore, the architecture and structure of said CNN can be seen in Figure 19.

Table 1 – Accuracy results for MLP and CNN models

		MLP (%)	*CNN (%)*	*REF (%)*
air1	Train	**99.95**	99.83	91.0
	Test	99.71	**99.8**	
clotheswasher1	Train	**99.98**	99.44	98.68
	Test	98.44	**98.83**	
dishwasher1	Train	**99.83**	99.76	96.854
	Test	98.31	**99.45**	
kitchenapp1	Train	**99.98**	99.86	99.647
	Test	**99.54**	99.43	
microwave1	Train	**99.95**	99.82	99.665
	Test	99.71	**99.75**	

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 72, 7, 8)	80
dropout_1 (Dropout)	(None, 72, 7, 8)	0
batch_normalization (BatchNo	(None, 72, 7, 8)	32
conv2d_4 (Conv2D)	(None, 36, 4, 16)	1168
batch_normalization_1 (Batch	(None, 36, 4, 16)	64
conv2d_5 (Conv2D)	(None, 17, 1, 32)	4640
batch_normalization_2 (Batch	(None, 17, 1, 32)	128
flatten_1 (Flatten)	(None, 544)	0
dropout_2 (Dropout)	(None, 544)	0
dense_2 (Dense)	(None, 40)	21800
batch_normalization_3 (Batch	(None, 40)	160
dense_3 (Dense)	(None, 5)	205

Figure 19 – Best CNN Architecture

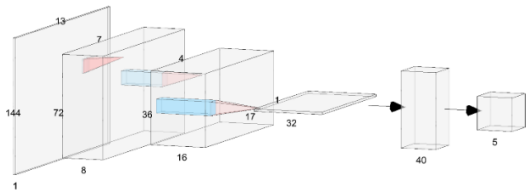

Figure 20 - Best CNN structure – AlexNet style [13]

As we can see in Table 1, the CNN offers comparable and sometimes better results for larger appliances like the AC, clotheswasher and dishwasher, but seems to perform worst in smaller ones like the kitchen circuit and microwave. This is mainly due to the fact that we are comparing a single CNN model versus 5 MLP models, and that the latter probably has some overfitting as no regularization techniques were applied to them. The reasoning behind this is that we wanted to establish reference results from simple and vanilla models.

To further analyse the results obtained from both models, a visual representation of the confusion matrix was made similarly to the visualization of the constructed output, explained in section III.C. For better comprehension purposes, the following figures will only contain days 30 to 200, and the legend for all these figures can be seen horizontally in Figure 21.

In this set of figures, we can clearly see that the CNN performs betters, giving less false positive that its MLP counterpart.

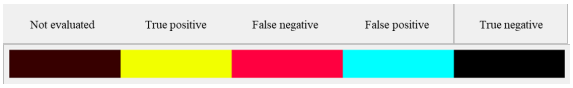

Figure 21 - Legend for graphic representation of confusion matrices

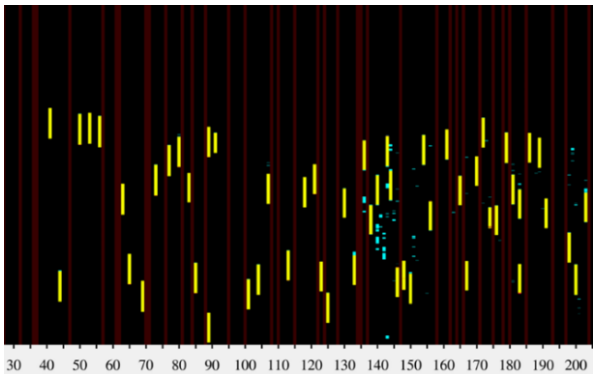

Figure 22 – Graphic representation of the confusion matrix MLP train set – dishwasher1

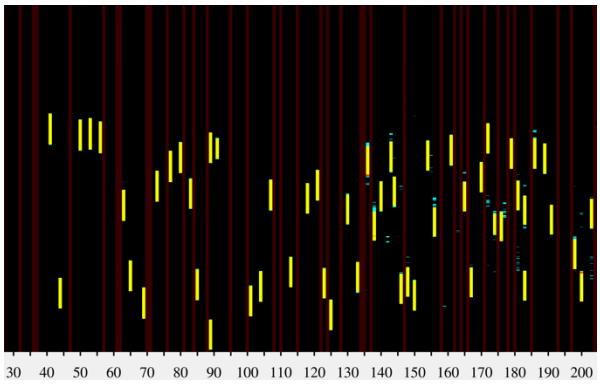

Figure 23 – Graphic representation of the confusion matrix CNN train set – dishwasher1

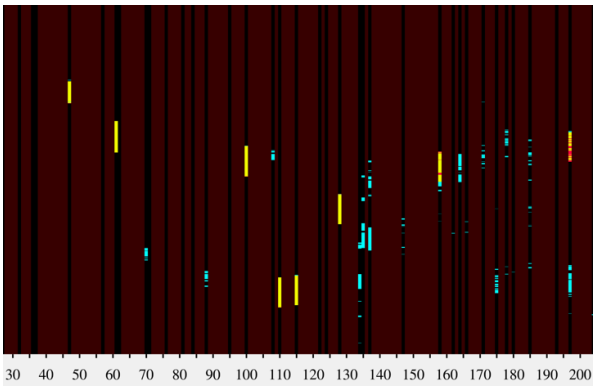

Figure 24 – Graphic representation of the confusion matrix MLP test set – dishwasher1

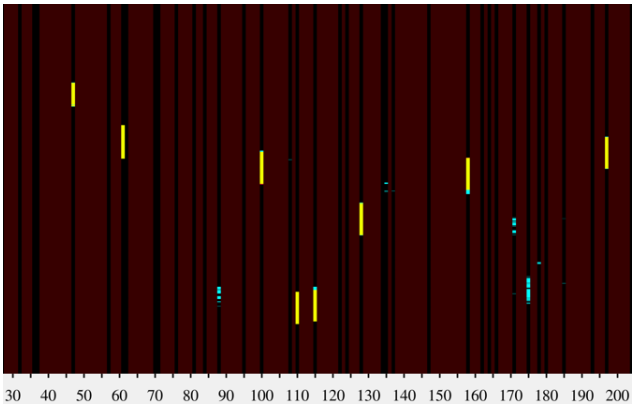

Figure 25 – Graphic representation of the confusion matrix CNN test set – dishwasher1

To further test the robustness and performance of the CNN model, we evaluated it on the data of another house of the same dataset (house with ID 2556). As we can see in Figure 26 and Figure 28 the results were not great. This was to be expected, mainly because the model was trained with the data of only a single house without any normalization of the input data, making it impossible for the model to generalize well outside of this data. As the data is heavily dependent of consumptions habits and the particular appliances of the house, it would be extremely improbable to find the same behavioral consumption patterns in two completely different houses.

To mitigate this problem and to check the generalization capability of the features obtained by our CNN, we used transfer learning from our trained model, freezing all the layers up to the first dense layer, and retrained the last two fully connected layers with the data from this second house. These results can be seen in Figure 27 and Figure 29. Observing these figures and comparing them with the results without transfer learning, we can see that the output is far better when applying transfer learning. This means that the features extracted by the CNN fitted with the first house are valid for the second one, proving these results quite promising.

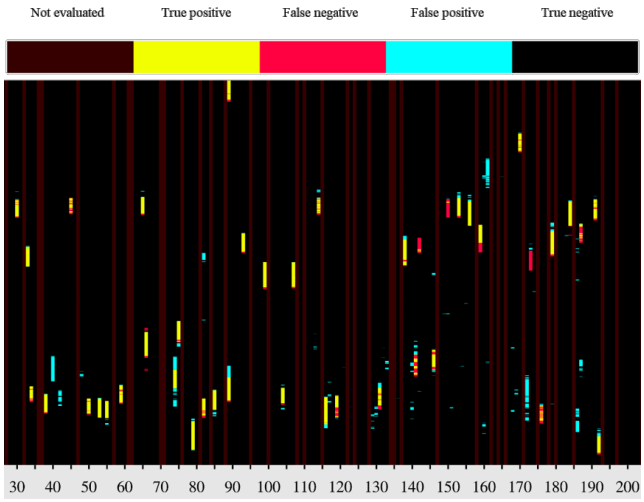

Figure 26 – Confusion matrix from second house with CNN –train set

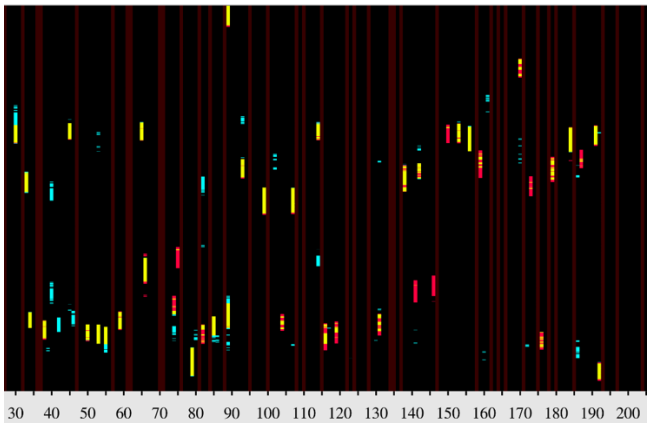

Figure 27 – Confusion matrix from second house with CNN and transfer learning – train set

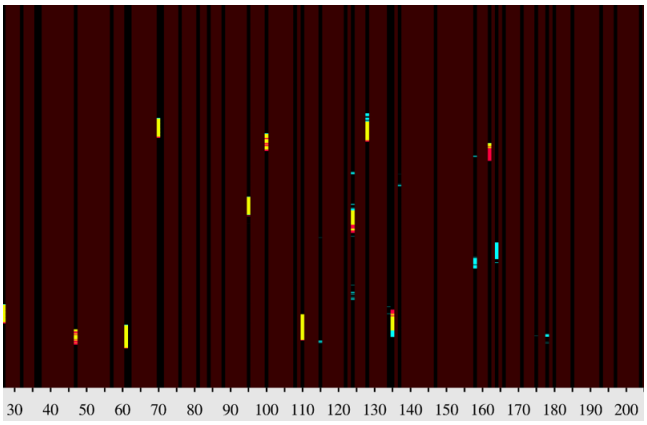

Figure 28 – Confusion matrix from second house with CNN – test set

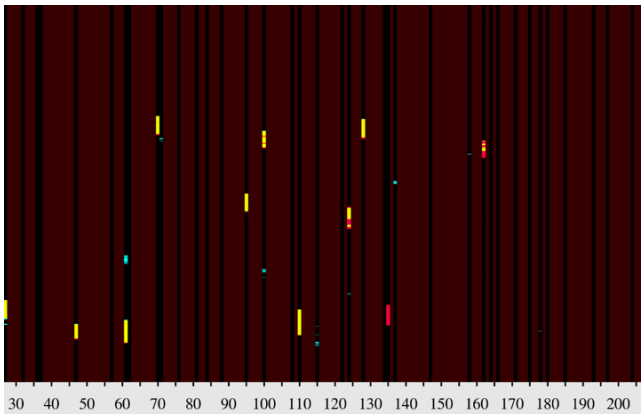

Figure 29 – Confusion matrix from second house with CNN and transfer learning – test set

Additionally, we can check what the CNN is looking for in vizgrams by stimulating it with a given vizgram and watching the output of the filter of each convolutional layer, in other words, the so-called feature maps. The vizgram chosen for this purpose can be seen in Figure 30. It consists of a dishwasher activation when no other appliances were on.

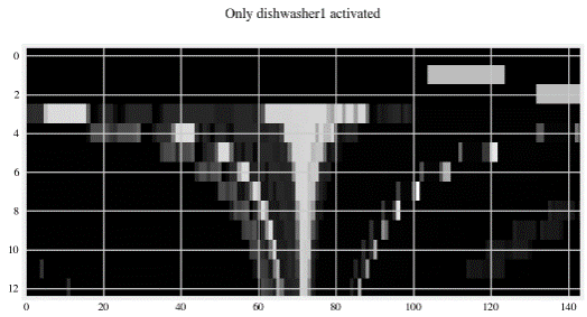

Figure 30 – Vizgram when only dishwasher1 was on

The feature maps of the first convolutional layer can be seen in Figure 31. We can see that this layer is looking for and extracting different aspects of the input. While several filters are looking at the temporal component of the vizgram, others are looking for the higher power peaks and a few others are looking for even more remote peaks. We can also view the feature maps of convolutional layers 2 and 3 (Figure 32 and Figure 33 respectively).

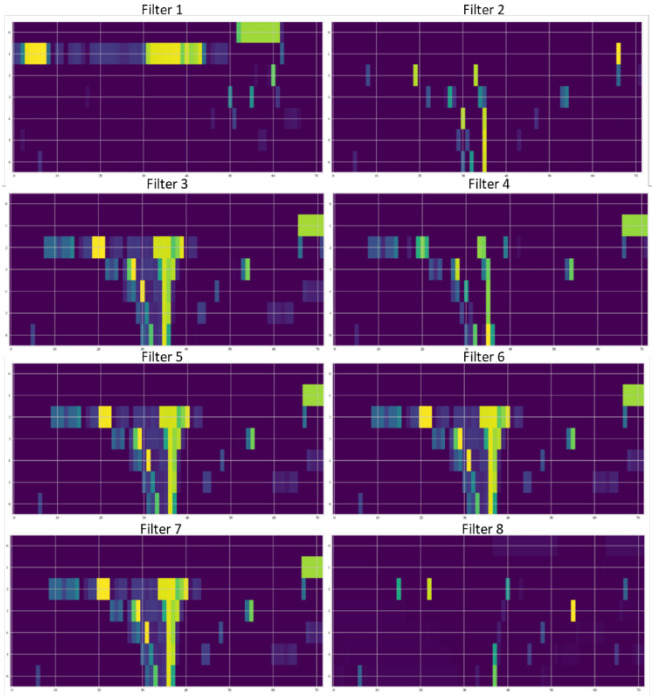

Figure 31 – Feature maps of first convolutional layer

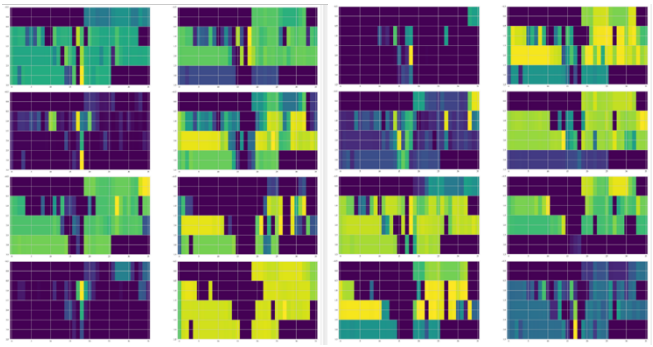

Figure 32 – Feature maps of second convolutional layer

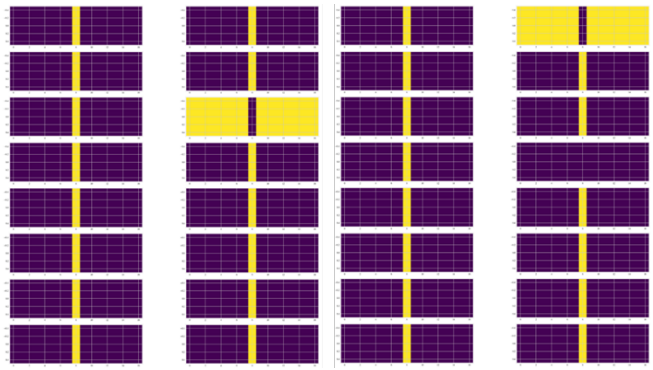

Figure 33 – Feature maps of third convolutional layer

In Figure 33 we can see that all the features extracted from the previous layers result in this barcode-like, single dimensional output, based upon which the fully connected layer of 40 neurons and the final output layer of 5 neurons will output the most likely appliance that was being used in that particular instant t_0.

Finally, to highlight an added value of this project, we created a series of compact and easy to read visualizations that give more information about the consumption habits of a given house. To plot some of them, we also focused on the number of activations of certain appliances. For example, in the case of the dishwasher and clotheswasher, knowing how many times any of those appliances were used throughout a week, or a month, or every Monday of every week, is much more useful that knowing exactly how many minutes it was on and off. This also increases the margin for error as being a couple of minutes off between the start of the appliance or the end of its use is not longer a problem, we only need to count a certain number of continued '1' as an activation. This was done for the aforementioned appliances, as the activations for the smaller ones were at most of 1 or 2 minutes. With the interval between timestamps being as small as 1 minute, the margin for error for these smaller appliances is enormous and invalidates this kind of analysis for them. We can nevertheless simply count the number of minutes they were on/off.

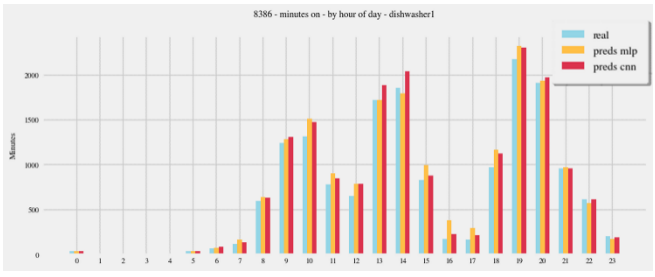

Figure 34 – Minutes on by hour of day – dishwasher1

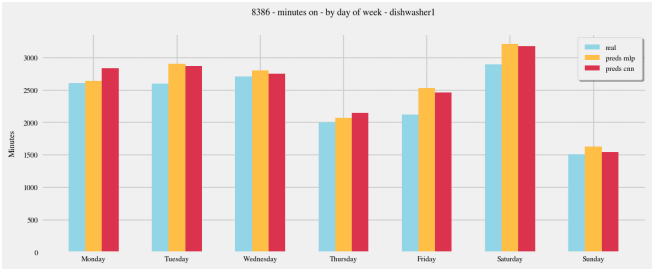

Figure 35 – Minutes on by day of week – dishwasher1

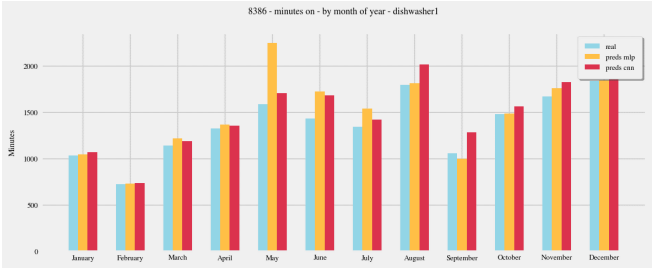

Figure 36 – Minutes on by month of year – dishwasher1

Figure 34 shows the minutes on by hour of day across all the year, comparing the real data, the predictions of the MLP and the predictions of the CNN. As we can see in this picture, results are very good, showing very little difference between both models and the real data, and between both models themselves. Figure 35 show the minutes on by day of week, and the results are once again very good. Finally, Figure 36 show the same data but by month of year, where, again, we can see little variance between the real and predicted data, except in May where we can see larger differences between the MLP and the real data.

These three figures also give us information about when is this appliance most used. For instance, by looking at the hourly grouped data, we can clearly see that in this particular house, the dishwasher is set mostly at 9-10, at 13-14 or at 19-20. From the weekly grouped data, we can clearly see that the dishwasher is rarely used on Sundays, and with the monthly grouped data we can infer that in February and September the occupants of the house probably away most of that month.

As for the number of activations, the data tells a similar story, perhaps even clearer. To compute this, an algorithm resembling the one described for filtering and filling was designed. This algorithm seeks every '1', starting a counter. Once it reaches a '0', it compares the counter with a certain threshold. If the value of the former is greater than the latter, we count those '1' as one activation. This is done for the real data, the predictions of the MLP and the predictions of the CNN.

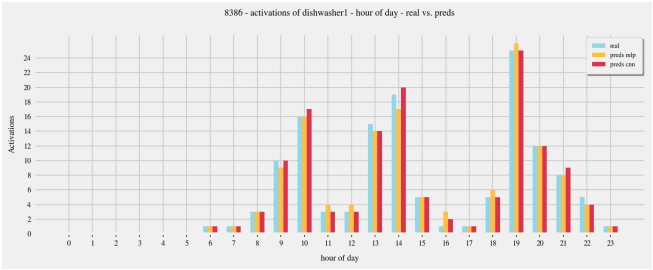

Figure 37 – Number of detected activations by hour of day – dishwasher1

Figure 37 shows the result of applying this algorithm, and we can see the same pattern as in Figure 34.

As explained earlier, these results are great for bigger, larger appliances such as the AC, dishwasher and clotheswasher, but the model shows its limitations when evaluated on the microwave or kitchen circuit. For instance, Figure 38 shows the minutes on for the microwave grouped by month of year. Here, we can clearly see that the CNN is not able to get good predictions, whereas the MLP does (even if it probably is overfitted).

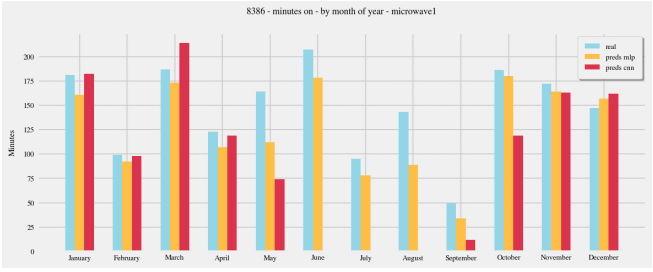

Figure 38 – Minutes on by month of year – microwave1

As a final way of evaluating our models, we also calculated the mean activation times for the dishwasher and clotheswasher. This was done by placing a value of -0.5 at the index where we considered an activation to start, and a value of 0.5 where we considered it to finish (to compensate when grouping afterwards). This time we will look at the data of the clotheswasher.

Figure 39 shows the mean activation time by day of week of the clotheswasher, and we can see that the CNN is again getting worse results than those obtained with the MLP.

This could be explained by how the activations were accounted for. We explained earlier that calculating these activations made less relevant the exact time of start/stop of the appliance. However, when calculating the mean activation times, we find ourselves in the opposite scenario, where the start/stop detection becomes all the more important. To see if outlier values were affecting heavily this mean, we also calculated the mean activation time, as is shown in Figure 40. However, as we can see in Figure 40, the results remain the same.

Observing the results from Figure 38 and the conclusions drawn from Figure 39 and Figure 40, it becomes clear that the CNN does not perform as well as the MLP, in the case of smaller appliances, in some particular months: the ones corresponding to summer. This period of time is precisely where the AC, an appliance with longer activation times and higher consumption, is used the most. This could be the main reason why the CNN model is not getting good predictions, as there is only one model for all appliances for the CNN vs. one individual model in the case of the MLP.

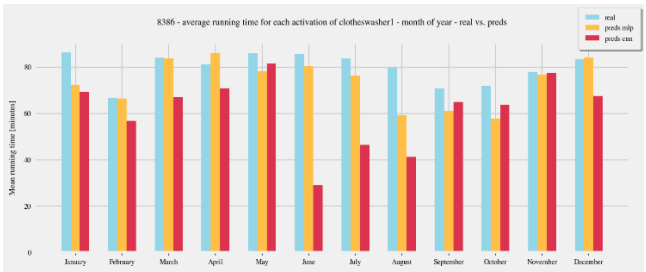

Figure 39 – Mean activation time by day of week – clotheswasher1

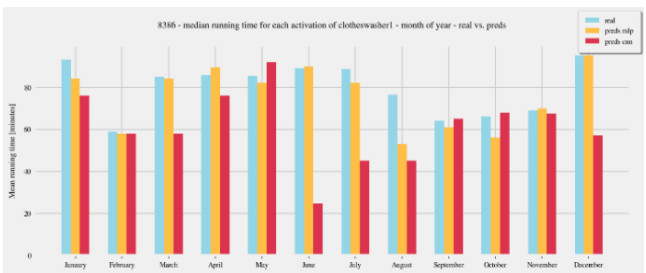

Figure 40 – Median activation time by month of year – clotheswasher1

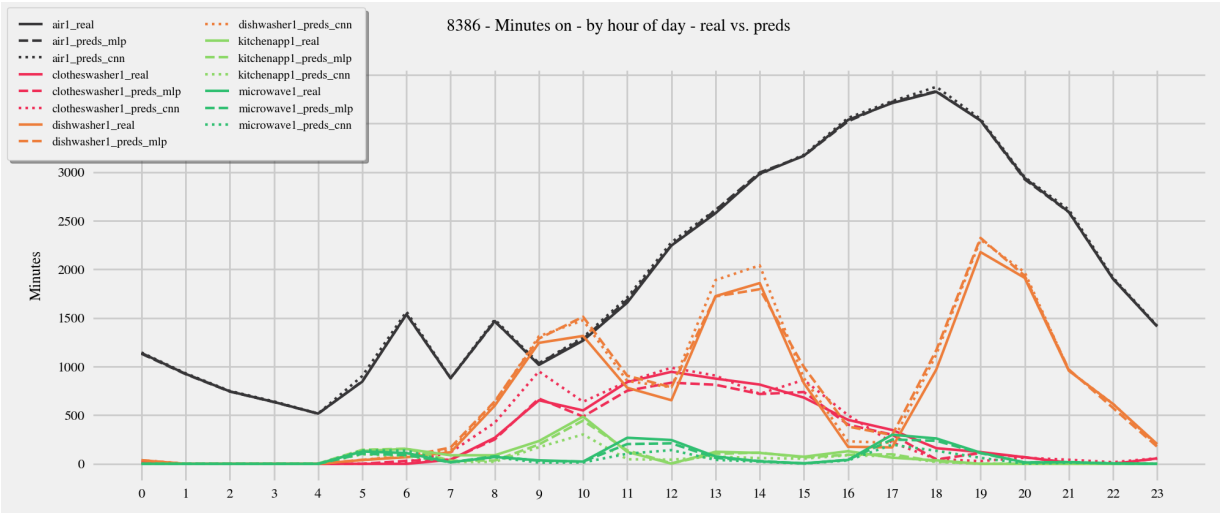

Figure 41 – Minutes on by hour of day – all appliances

Finally, and to sum up the results, Figure 41 shows the minutes on for all the appliances and for the real data, MLP and CNN models all together. This figure shows how results from both models are great and robust, offering very little difference between the real data and the predicted data.

V. Conclusions and Future Work

This work explained a new way to process the data that serves as input for Machine Learning algorithms in the field of NILM. The vizgram, a visual representation of a matrix that contains information of the instant it is sampling as well as information from the past and the future, within a certain timeframe.

Results show that this information is useful and be used directly as matrix data for MLP models as well as visual data for CNNs. It is important to note that the results obtained with the MLP were from 5 different MLP models, one for each appliance, whereas the results from the CNN were obtained from a single CNN classifier. Seeing as the results were similar, one can safely say that the CNN was superior in the task of identifying the on/off activation of appliances, meaning the vizgram was a useful representation of the data, as it was also demonstrated when seeing the feature maps of the convolutional layers and the potential of a quick transfer learning from our own model.

It is also important to note that the construction of the input data is extremely modular and easy to modify. Normalizing by the maximum data or by the contracted power, changing the different parameters defining the vizgram or further processing the construction of the output are some of the possibilities this paper did not cover and that could yield better results. Of course, it goes without saying that the best way to improve an ML problem is to train with more data, which we could not do due to hardware restrictions

As in any machine learning problem, training with more data is also one, if not the best solution to achieve better results. Mixing the training data with different behavioral consumption patterns from different households would probably result in a better and more robust model.

Finally, the last section shed some light on how this data can be useful in a more professional sector. Knowing how many times the appliances were used, by hour of day, day of week or month of year with only the total power consumption as input is extremely relevant when designing new targeted products and offers. It can also serve as data for recommendations on how and when to use appliances, resulting in a lesser cost if the power is shifted, while reducing power consumption waste altogether.

References

[1] A. Zoha, A. Gluhak, M. Imran, and S. J. S. Rajasegarar, “Non- intrusive load monitoring approaches for disaggregated energy sensing: A survey," Sensors, vol. 12, no. 12, pp. 16838-16866, Dec. 2012.

[2] G. W. Hart, “Nonintrusive appliance load monitoring”. *In proceedings of the IEEE* 80 (12):1870–1891, Dec. 1992.

[3] J. Froehlich, E. Larson, S. Gupta, G. Cohn, M. S. Reynolds, and S. N. Patel, "Disaggregated End-Use En- ergy Sensing for the Smart Grid". In proceedings of Pervasive Computing, IEEE 10 (1):28-39, Jan. 2011.

[4] S. Barker, M. Musthag, D. Irwin, and P. Shenoy, “Non- Intrusive Load Identification for Smart Outlets”. In pro- ceedings of the IEEE International Conference on Smart Grid Communications (SmartGridComm):548–553, Nov.

[5] J. Gao, E. C. Kara, S. Giri, and M. Berge´s, “A feasibility study of automated plug-load identification from high- frequency measurements”. In proceedings of the 3rd IEEE Global Conference on Signal and Information Processing (GlobalSIP):220–224, Dec. 2015.

[6] K. Ehrhardt-Martinez, K. A. Donnelly, and S. Laitner, “Advanced metering initiatives and residential feedback programs: a meta-review for household electricity-saving opportunities,” 2010: American Council for an Energy-Efficient Economy Washington, DC.

[7] S. Darby, “The effectiveness of feedback on energy consumption,” A Review for DEFRA of the Literature on Metering, Billing and direct Displays, 2006.

[8] Benchmarking smart metering deployment in the EU-28. December–2019

[9] Bouhouras, A.S.; Gkaidatzis, P.A.; Chatzisavvas, K.C.; Panagiotou, E.; Poulakis, N.; Christoforidis, G.C. Load Signature Formulation for Non-Intrusive Load Monitoring Based on Current Measurements. Energies 2017, 10, 538.

[10] Kim, H., Marwah, M., Arlitt, M., Lyon, G., & Han, J. (2011). Unsupervised Disaggregation of Low Frequency Power Measurements. *SDM*.

[11] Puente, C.; Palacios, R.; González-Arechavala, Y.; Sánchez-Úbeda, E.F. Non-Intrusive Load Monitoring (NILM) for Energy Disaggregation Using Soft Computing Techniques. *Energies* **2020**, *13*, 3117.

[12] https://www.pecanstreet.org/dataport/

[13] http://alexlenail.me/NN-SVG/LeNet.html