 doi:10.20944/preprints202201.0313.v1

Event-Based Clustering with Energy Data

Kieran Greer and Yaxin Bi,

School of Computing, Faculty of Computing, Engineering and the Built Environment, Ulster University, UK.

Email: K.Greer@ulster.ac.uk, kgreer@distributedcomputingsystems.co.uk; Y.Bi@ulster.ac.uk

Abstract

This paper describes a stochastic clustering architecture that is used in the paper for making predictions over energy data. The design is discrete, localised optimisations based on similarity, followed by a global aggregating layer, which can be compared with the recent random neural network designs, for example. The topic relates to the IDEAS Smart Home Energy Project, where a client-side Artificial Intelligence component can predict energy consumption for appliances. The proposed data model is essentially a look-up table of the key energy bands that each appliance would use. Each band represents a level of consumption by the appliance. This table can replace disaggregation from more complicated methods, usually constructed from probability theory, for example. Results show that the table can accurately disaggregate a single source to a set of appliances, because each appliance has quite a unique energy footprint. As part of predicting energy consumption, the model could possibly reduce costs by 50% and more than that if the proposed schedules are also included. The hyper-grid has been changed to consider rows as single units, making it more tractable. A second case study considers wind power patterns, where the grid optimises over the dataset columns in a self-similar way to the rows, allowing for some level of feature analysis.

Keywords: stochastic clustering, energy prediction, disaggregation

1 Introduction

This paper describes some methods for clustering and making predictions over energy data, using a stochastic Hyper-Grid [9][19]. The topic relates to the IDEAS Smart Home Energy Project [11], where a client-side Artificial Intelligence component can predict energy consumption for appliances. The proposed data model is essentially a look-up table of the key energy bands that each appliance would use. Each band represents a level of consumption by the appliance. This table can replace disaggregation from more complicated methods like probability theory, for example. Disaggregation is the process of estimating how much energy each appliance would use from a single input amount. It is helpful to be able to estimate this, so that it can be learned by an AI model that can then make future predictions over similar appliance usage. Because the energy footprint for an appliance is quite unique however, the

 doi:10.20944/preprints202201.0313.v1

energy bands generated from it are also quite unique, which may be enough to identify each appliance, rather than rely on more complex probability methods. The energy provider therefore, is tasked with trying to predict how much energy will be required at a particular time. The provider sees this as a single problem over the whole set of input variables. The user-side is more discrete, when there can be more than 1 independent entity, resulting in input that is more event-based. As such, it may be more appropriate to split the client-side model into separate parts, each modelling one of the independent entities. A stochastic method that can produce non-continuous solutions may therefore have some advantages over a functional model. A Hyper-Grid is therefore proposed for clustering the data and can accommodate this essential difference - not the single provider, but discrete consumers with unrelated events. Then to aggregate the clustering results, the Frequency Grid [8][19] can be used.

The process is stochastic in nature, clustering randomly, only a subset of the input data each time. It is costly to run, but this can be configured with smaller-sized dataset batches, when the results from each batch can be accumulated and so it can in theory be used with larger datasets, but over a longer period of time. The tests firstly consider creating a lookup table of unique energy bands to describe each household appliance. While this should be very quick to use in real-time, it is also able to provide a reasonable amount of economy, by making the energy prediction more accurate. The tests are then extended to consider clustering feature sets in the row clusters. As such, the Hyper-Grid can cluster rows first and then features of those rows, using the same method. Tests on energy data show that the method can produce consistent and logical results and has potential for a wide range of applications.

The rest of the paper is organised as follows: section 2 gives some related work in the field of energy prediction and what other methods are typically used. Section 3 then summarises the new clustering algorithms. Sections 4 and 5 consider the two case studies and describe the test results. Finally, section 6 gives some conclusions on the work.

2 Related Work

2.1 Stochastic Clustering

Stochastic clustering is not new and in fact it may be the preferred method for clustering something like electric vehicle charging [20]. They state that coordinated charging of EVs can bring some benefits by itself and implementation of this scenario without coordinated charging can impose a huge amount of excess load on the grid. They give a description of stochastic scheduling, where they state that ‘since there are a lot of uncertainties in a real-world situation and specifically in EV energy scheduling problems, such as EVs driving pattern, diverse temporal and spatial EV charging pattern and so on, in the literature in this field, deterministic scheduling has been implemented much less than stochastic one. A stochastic model has one or more stochastic elements. The system having stochastic element is generally not solved analytically and, moreover, there are several cases for which it is difficult to build an intuitive perspective.’ While this paper is interested in a different field of energy prediction, it has to solve a similar type of problem.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2022 doi:10.20944/preprints202201.0313.v1

To make the Hyper-Grid practical, it is possible to use a form of bootstrapping over randomised datasets. Bootstrapping [7] removes some rows from the dataset each time and solves the problem for the remaining rows. It then averages over the final solutions. The hyper-grid by nature, solves over localised parts of the whole problem and so this is always part of constructing any solution. It would therefore be natural to include a bootstrapping process with the hyper-grid, which would allow it to use subsets of the whole dataset each time. It would also help a lot in practical terms, to reduce the size of the data grid, which is very time-costly to solve. This method is also the one proposed in [3], that suggests protecting linear classifiers from adversarial attack, through the use of bagging and random subspaces. Aggregating the subset results from the hyper-grid is therefore a very similar idea. The architecture may also have similarities with neural networks. Deep Learning neural networks [5], for example, learn discrete elements before summarising them through a pooling layer. The stochastic and distributed nature however may have more in common with random neural network architectures [18]. Likelihood estimators have been used to predict energy usage before. One new version associated with Generative modelling (GANs) and variational autoencoders (VAEs) is described in [17]. Part of their intuition states:

'Since likelihood is the product of densities evaluated at all data examples, the model density at each data example should be high. Suppose we don't observe the model distribution directly, and instead only observe independent and identically distributed (i.i.d.) samples drawn from the model. Because the density at data examples is high, more samples are expected to lie near data examples than elsewhere.'

It then defines a series of likelihood estimates for random variables that contain distances between *x* and the nearest sample and uses the minimum solution as the best one. Another paper [4] uses regression models to predict the appliance energy usage in a house. It notes that the larger white goods appliances (fridge, cooker, clothes washer, freezer) consume the most energy and appears to suggest that their best regression model can predict the energy consumption to 57% accuracy. If that is the case, then the results of the case study in section 4 are not too bad. Another paper that uses genetic algorithms to predict energy usage [13], quotes 29% saving or 36% during peak time.

2.2 Energy Systems

Predicting the household energy use at the level of applications is quite a popular topic. Modern systems are quite inefficient and so there is a potential for a huge amount of saving. Because it is difficult to measure each application in real time and it is also an intrusive and possibly expensive process, the current trend is to disaggregate the total power input to each appliance, to estimate their use from a behaviour model, generated from AI. This is typically done by first using raw data to train an AI model of the appliances and then using that to estimate the disaggregated values for each appliance. The training phase would typically log on-off switching events for each appliance in the house and then train possibly a Hidden Markov Model to recognise the hidden or unobserved states, which are the individual appliances, from the observed state, which would be the total power usage per unit time [1][2][14][16]. After training, the internal measurements do not need to be made, but can be

 doi:10.20944/preprints202201.0313.v1

estimated by the model. While this system is shown to work well, it is proposed in this project that it is overly complicated and that with today's computing power and memory resources, a much more direct approach could be just as effective. For one thing, the Markov Model is state-based and time-related, where switching events are placed in order. The new method to be proposed is set-based, where it will suggest an amount of energy that may get used during the day, but not the exact order in which it gets used. The set-based approach has been looked at previously, where gaussian equations are used as part of the model and the Markov process [2] can be an extension of a Bayesian Network [15] for example. But these calculations are also expensive and may also include other probability measures, such as Expectation-Maximisation or Likelihood. The method in this project will make use of the idea that most appliances have quite a unique footprint, in terms of their unit energy use, and so a large lookup table may be sufficient to allow disaggregated appliance usage to be recognised from a single input power measurement.

3 Summary of the Clustering Algorithms

3.1 Hyper-Grid

The Hyper-Grid [9][19] creates solutions by matching data rows that are closer together depending on some measurement, where one idea is to evolve the matched pairs further, as in Genetic Algorithms [10]. For this project however, it forms the basis for some clustering phases that result in energy bands, which can define the appliance usage over the course of a day. The heuristic works as follows: It reads a dataset of values, randomises the row ordering and also keeps a record of the original ordering for identification purposes. For the problem of learning the appliance behaviour, the dataset is a single column of values, representing the appliance energy consumption every time unit, but the dataset could have any number of columns. Randomising the row ordering means that there cannot be any bias in the ordering during the row matching process. The heuristic then compares the data rows and notes which pairs are most similar, according to some metric, such as the Manhattan distance. So that the heuristic does not result simply in hill-climbing, a matching process is preferred to one that selects the largest scores only. Also, if two rows are selected, any rows between them are removed from the solution, which means that it also discriminates. All of the potential matches are saved and then sets of row pairs are selected that would optimise the total score. This optimisation gives the heuristic some direction and helps to ensure a better result. The algorithm is illustrated in Figure 1, where the process would be as follows:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2022 doi:10.20944/preprints202201.0313.v1

Row Index	Column	Column	Column
1	3	5	3
2	2	3	3
3	4	4	3
4	5	3	5
5	5	4	3
6	6	3	5
7	2	4	1

Figure 1. Example matching process by the Hyper-Heuristic.

1) Compare every row with every other row and save a difference score based on the Euclidean difference between the cell values.
2) The rows that are closest to each other are:
   a) 1 and 7 with difference 4
   b) 2 and 7 with difference 3,
   c) 3 and 5 with difference 1,
   d) 4 and 6 with difference 1.
3) The sum score for rows 1 and 7 is 18 and for rows 2 and 7 it is 15. But rows 2 and 7 are more similar and so they are preferred.
   a) The rows in-between can be removed, but if they also have matches, then the matches can be added first.
4) The similarity scores for rows 3 and 5, or 4 and 6 are the same with a score of 1. The sum score for rows 4 and 6 is larger however, with a value of 27 and so rows 4 and 6 are preferred. Because this cuts across rows 3 and 5, the matching pair of 3 and 5 is not included in the solution – row 5 is removed from the solution.
5) This leads to a final solution with the row pairs 2 and 7, and 4 and 6.
6) These row pairs are then saved to a list that can store all rows pairs for all of the test runs.
7) The final list of row pairs are then fed through the frequency grid that clusters them into mini-clusters for similar frequency counts.
8) The mini-clusters can then determine the energy bands, for example.

For the appliance problem and again to prevent hill-climbing from the unique footprint, row matches can be treated as equal if the difference falls inside of a particular value or band and is not only the smallest difference possible. Therefore, if a band similarity value is 2 and one matching score is 0 and another is 2, then they would both return a band value of 1. The use of bands or score ranges is interesting and might also work with other algorithms.

Due to combinatorial explosion, a complete search over a larger dataset is not possible and so the algorithm has to split a dataset up into smaller-sized parts and solve the problem on each part separately. Because the rows are randomised first, this can still give a reliable result. For example, if Figure 1 is a subset of the whole dataset and there are another 7 rows that

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2022 doi:10.20944/preprints202201.0313.v1

have been clustered during a different batch run, the row pairs from the other subset can be added to the row pairs from Figure 1 and the combined list can be clustered using the frequency grid.

3.2 Frequency Grid

The Hyper-Grid therefore also requires an aggregating layer, which can be the Frequency Grid [8][19], for example. This reads the list of row pairs and produces sets of count values that represent which rows are more often paired together. It is more entropy-based than local counts however, where the aggregation from the frequency grid can produce a holistic view of the row pairs and produce clusters for the whole dataset. The following description is taken from [8]. Consider a different set of data, but again for 7 rows, shown in Figure 2.

	1	2	3	4	5	6	7
1	x	4	4	4	2	1	1
2	4	x	4	4	1	0	0
3	4	4	x	4	1	0	0
4	4	4	4	x	1	0	0
5	2	1	1	1	x	3	3
6	1	0	0	0	3	x	3
7	1	0	0	0	3	3	x

Figure 2. Frequency counts would group rows 1-4 and rows 5-7 together.

It is clear from the data that rows 1-4 all reinforce each other (pattern 1), as do rows 5-7 (pattern 2). With a grid format, the input is represented by a single pattern group, where a count is incremented for each row pair occurrence. The grid format lists each variable, or in this case it would be a row number, both as a row and a column. Each time a pattern is presented, the related cell value for both the row and the column is incremented by 1. In row 1, for example, the counts suggest that it should be clustered with rows 2-4, because they have higher counts with row 1. The same conclusion can be made for rows 5-7. It is probably not necessary to update a self-reference in the grid, so the leading diagonal can be empty. The grid result can then be read using the following algorithm:

1) Each row displays count values representing a key variable - the row name, and its relation to the other variables.
2) All cells relating to variables in the input pattern are updated each time.
   a) Each row that starts with one of the variables updates the count for every other variable in the input pattern.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2022 doi:10.20944/preprints202201.0313.v1

b) Because the variable is repeated in several cells, this still leads to normal count values for each cell.

3) To determine the best clusters then:
   a) For a key variable (row key value) scan across and select the other variables with the largest count values. That variable then considers those other variables to be part of its cluster.
   b) The other variables however may be more associated with a different cluster, so their rows can be checked for consistency. They should similarly have a largest count value for the other variables in the cluster. If any have different (larger or smaller) count values, then they probably belong to a different cluster..

3.3 Feature Selection

A second use for the algorithm is suggested that is a self-similar process of matching over the columns instead of the data rows. This could work as follows and is the basis for the second case study: The algorithm is run on the dataset to produce sets of mini-clusters. These are subsets of larger clusters that are realised from subsets of the whole dataset. Then for each mini-cluster, the rows and columns are transposed and the algorithm is run again on that subset only, to produce another set of mini-clusters that would define feature sets instead. Some of the mini-clusters may share some of the feature sets, when they can be combined further, depending on some threshold criterion. This process therefore also provides a basis for analysing the column or feature sets in the data, which is described more in section 5.

4 Case Study 1: Disaggregating Energy to Appliances

This was the main focus of the research on the IDEAS project [11][12] and it is a well-known problem of trying to predict how much energy is required, by measuring the energy consumption of a set of household appliances and using that model to make the prediction. The single input power source needs to be disaggregated to each of the appliances, or an estimate of how much would go to each appliance needs to be made. This problem therefore requires a training stage to learn the model and then a testing stage to match that with the input source. The proposed algorithm ran a number of bootstrapped tests on raw data and generated row pairs that resulted in sets of mini-clusters. As the clustering involves a similarity measure, a more obvious approach is to aggregate the raw data into the time unit, hours for example, and then count the number of occurrences of each aggregated value. If this is done however, there is too much variability in the aggregated values. They do not conform to a set of values and so it is not possible to produce an aggregated view. This may be because the behaviour of any appliance is unpredictable and so aggregated values will typically be different to each other. Therefore, some form of clustering is required to recognise patterns or structure in the data and the hyper-grid was selected for this project.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2022 doi:10.20944/preprints202201.0313.v1

4.1 Second Clustering phase

The first clustering phase therefore selected time slots during the day that were similar. A second clustering stage then took the full list of mini-clusters and retrieved the energy values for each row in a cluster and placed that in an energy cluster for the row cluster. Energy clusters that overlapped could result in an energy band with an upper and a lower limit, although, single values were more typical. These energy bands could then be used to determine the user behaviour of the system. For example, if there were 3 events of band *A* and 1 event of band *B*, the energy supplier could expect band *A*, 3 times more often. After band *B* occurred, it could expect only band *A* until it had occurred 3 times, and so on. The bands are most useful for reducing the complexity of the system and recognising some inherent structure or behaviour. If there is a range, then the upper limit would typically have to be accommodated for by the system. But it is the fact that there is now some sort of model that can describe the application behaviour in a tractable way that makes it useful.

4.2 Third Clustering Phase

A third clustering phase is also likely, not when training the data model, but when reading it into the energy network, to be used by the system during runtime. This would help to reduce the complexity even further, so that a lookup table can be generated for an exhaustive combination search. To do this, energy bands with the same 'integer' upper and lower-limit parts were further combined, resulting in only one energy band for the whole range. See *Table 1* in section 4.5, for an example.

4.3 Optimisation

The intention is to optimise some process that is part of an energy system. As described earlier in this section, the energy bands can be used to predict how much energy the appliances are likely to use. The system can therefore plan for the calculated maximum energy requirement at any one time and then when energy band events occur, they can be removed and the prediction can be adjusted to the remaining set. It is always important that there is enough energy provided for any eventuality and so there always needs to be additional energy in the system that may not get used. This is one place where energy savings can be made, if predictions can minimise this additional amount. Other methods may typically recognise on-off switching events for appliances and then generate Markov Models [1][2] or other probability measures, such as Likelihood evaluations [17] and with that research, disaggregation is helpful. The model would be trained to recognise on-off switching events for appliances, which gives a state-based or a probability model for each appliance. Then from a single input value, the system would disaggregate the input power source to each of the appliances, by learning when the appliances are likely to be on or off. A Markov Model can be used with time-based events, or a Likelihood probability estimate can be used with set-based events, for example. The method of this paper does not have sequential events, but is more set-based. It only defines that this set of events may have occurred in a particular time period. This is both good and bad, because it may be possible to produce an alternative to the Markov

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2022 doi:10.20944/preprints202201.0313.v1

Model, that is more flexible, but also more simplistic than a Likelihood estimate. On the down-side, it would still benefit from the accuracy of learning some amount of timing and so the future work for this section describes how that might be achieved.

4.4 Test Case

An algorithm has been implemented in the Java programming language and tested on real data [6] from households in Switzerland. This data has been used before to generate Markov Models, for example [1][2]. A number of houses were monitored, where a smart plug was able to measure the energy consumption for an appliance. This was relayed to a central system and logged every second. Therefore, each appliance was logged every second for a period of approximately 8 months. For the hyper-grid, a measurement of every second was too fine-grained and so the data was converted into aggregated values for each hour. That is the average amount of energy used by the appliance each hour. The data for each appliance was then clustered using the Hyper and Frequency Grids, as described earlier in this section. The final set of energy bands would be used to define the behaviour of the appliance over the course of a day. For the 8-month period, there was no large change in the appliance behaviour from one month to the next, but this could certainly be modelled as individual and seasonal sets of bands.

The energy system was then able to guess how much energy each house was likely to require during the day. It would have to provide the upper limit each time, so that the house does not run out of energy. This can be achieved by returning the largest energy band for each appliance in the house each time. While that is an upper limit, the system would test the accuracy of this by then removing an energy band instance, at random, for each appliance. The next prediction would then be calculated on the remaining energy bands and the accuracy would be the difference between the estimate and the randomly selected set of used bands. This system therefore does not need to model exactly when an appliance was used, but knowing when, would still make it more accurate.

4.5 Lookup Table

It is proposed that a simple lookup table can be used to good effect. The energy bands reduce the combinatorial complexity enough to suggest that for all appliances in a household, a lookup table of under 1 million entries could be sufficient. The table needs to map every energy band events for each appliance with every other one and so it has to provide a combination for every possibility. There could of course be statistical methods to reduce the number further, when the entries are very similar, such as the third clustering phase of section 4.2. For example, *Table 1* is a set of energy bands that were produced for a fridge in some household over the course of a day. Each band represents an hour of energy consumption by the appliance and the frequency is how many times that occurred during the day. Most bands contained a single value that was the result of the first two phases of clustering. In rarer occasions, for example: 31.0813 - 28.1024, there was an overlap in the clusters leading to a value range. When reading these into the model however, the lower-values bands with the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2022 doi:10.20944/preprints202201.0313.v1

same integer number can be combined. For example, all bands starting with the integer value of '1' can be combined.

Energy Band	Freq per Day	Energy Band	Freq per Day
51.3304 - 51.3304	1	12.3664 - 12.3664	1
47.3313 - 47.3313	1	8.2406 - 7.89404	2
41.8416 - 41.8416	1	6.6794 - 6.6794	1
37.4993 - 37.4993	1	4.0358 - 4.0358	1
35.5428 - 35.5428	1	3.8319 - 3.8319	1
31.0813 - 28.1024	5	3.2404 - 3.2404	1
21.3339 - 21.3339	1	1.3989 - 1.3989	1
18.3959 - 18.3959	1	1.2865 - 1.2865	1
16.317 - 16.317	1	1.2065 - 1.2065	1
14.8694 - 14.8694	1		

Table 1. Example of energy bands for an appliance, with power consumed per hour. Set of energy bands that occurred over the course of a single day for a Fridge

Then with each combination of all appliances, a power input total can be calculated by taking the upper limit or average of each energy band in the combination. The lookup table would be produced for each appliance during a short training phase, when the system is being setup. After that, the system would read the power consumption at some time unit and pass that to the table, which would return the appliance combination that matches closest to the power value. Because the band values are quite unique, the combination value can be a key to a table, where the table value is then the set of appliances that created it. The selected band events could then be removed from the day's set, allowing the next consumption amount to be more accurately predicted.

This method should work reasonably well because of the unique energy footprints, but there is also quite a wide margin of error that would be acceptable. For example, if the system has a set that contains energy bands of '5 units' for both appliances *A* and *B*. Then maybe the next event reads an energy band of 5 and the system mistakenly attributes that to appliance *B* instead of *A*. For the following time unit, the system would then expect appliance *A* to produce a 5-unit band instead of *B*. But if *B* produces the 5-unit band in the next time unit instead, this does not in fact harm operation of the system. The system only needs to match with the energy requirement, it does not need to know exactly, which appliance any band came from. Although, the authors recognise that for a more sophisticated system, individual appliance usage may need to be monitored and may prove more problematic.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2022 doi:10.20944/preprints202201.0313.v1

4.6 Test Results

The results show potential for improvement, but the path to that improvement is outlined in section 4.7. After the energy band clusters were produced, an energy network and disaggregator were created from them and a predictor was asked to simulate the network activity. It would return its maximum requirement for energy each time and then remove a random set of energy bands as the actual event. The maximum requirement could in fact be summed and the total then passed through the disaggregator, to return an estimate for it instead. This produced only a small reduction in the accuracy overall. For 3 locations of the datasets [6], the following result of *Table 2*, was achieved.

Total power consumed was: 55295 Difference between predicted and disaggregated was: 6.0E-12, or 1.0E-14 % Difference between predicted and used was: 44599, or 80.5 % Number of predictions less than actual was: 0 Saving for location: h1 was 14414 or 54 %, from 24 events. Saving for location: h2 was 13330.5 or 28.5 %, from 24 events. Saving for location: h4 was 2222.5 or 18.5 %, from 24 events.

Table 2. Prediction accuracy for locations House 1, 2 and 4.

The first figures indicate that the accuracy of the prediction to the actual random events is only about 20% accurate, or there is an 80% power loss when predicting how much energy should be provided. But this is for random selections that have to accommodate the spiking events. Disaggregating the single input source to the appliance lookup table however is very accurate and the error may be down to the computer processing floating point numbers. This gives support to the idea of unique energy bands for the appliances. Then, the difference in providing the upper bound on the energy bands each time and the predicted amount, over the course of a day, is shown in the second set of figures. It could be around 50% savings, but house 4 is less at only 19% savings. A key concern is to guess when the spiking event might occur. This is where an analysis that includes time would be helpful. A calculation using only the upper bound would have to accommodate this for every hour, while the prediction can remove it as soon as it occurs, which on average might be half-way through the day for random events, for example. But the energy bands themselves are a unique solution that make the whole problem very tractable.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2022 doi:10.20944/preprints202201.0313.v1

4.7 Future Work

Tests suggest that the system can save maybe 50% of energy production from providing the maximum amount each time, but that it is still relatively inaccurate at predicting the random events. Until the spiking event has occurred, the system has to accommodate it and so it has to provide that level of cover, when the appliance is only using a small amount of energy. This is therefore where a set-based approach is much less effective than one with event sequences. The IDEAS [12] system can also provide appliance schedules however, entered manually by the user, that can describe in general terms, when an appliance is likely to be used. These schedules can therefore describe time-based events that could be used as part of the prediction. For example, if the schedule indicates high usage in the morning and low usage in the afternoon, then this can be added to the prediction. A second area of interest is the energy sources and their power input. This is disaggregated as part of the system operation, but the power input is also often modelled as part of a time series. The time series would then produce timed events that could also be used to adjust the prediction.

5 Case Study 2: Germany Wind Power Regions

A second test scenario looked at clustering the Wind Power datasets from Germany [21]. This data was selected because it is relatively small in size and does not require formatting. There are 4 datasets, one for each of the four wind power companies - 50Hertz, Amprion, TennetTSO and TransnetBW. Each row in the dataset represents 1 day, over a 12 month period, resulting in 397 rows. Each column then represents the power generated each 15 minutes of the day, resulting in 96 columns. The analysis differences were quite small, but they may represent some kind of pattern. Also and importantly, they may help to confirm that the hyper-grid is able to produce consistent results.

Two sets of tests were run. The first was for 200 iterations over the dataset batch subsets and the second was for 500 iterations over the same-sized batch subsets. For example, if the batch size was 100 rows, then there would be 4 batch tests for every iteration. The first test resulted in mini-clusters that covered only about 10% of the whole dataset, while the second test covered about 50% of the data rows. It might be expected that a smaller number of rows would select slightly larger values, because of the optimising feature in the algorithm. If the data is represented by some distribution, for example, then the smaller number of rows may be more from the top of that distribution. Then as more rows are selected, they will include rows from lower down the distribution, or with smaller values. This actually means that overtraining of the hyper-grid would probably result in the clusters joining with each other and losing some of the discrimination effect.

Therefore, some configuration was required, to select an appropriate size of batch subset, a similarity percentage for row matches, and so on, which was a bit of a black art. A best configuration would also depend on the data itself, as to how many rows would eventually be included. When the rows had been clustered, each mini-cluster was analysed further to match its columns or features and in this case, bands were not used, when the closest match value was exact instead. The features were not studied in detail, but only the variance was

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2022 doi:10.20944/preprints202201.0313.v1

considered, which is OK, because they represented time units only. Row clusters that shared a certain number of similar feature clusters were then combined again afterwards, during the analysis stage, described next.

5.1 Dataset Analysis

The test program therefore, produced sets of mini-clusters, or row indexes that should be clustered together and for each mini-cluster it did the same for the columns. What might be interesting was the fact that the column clustering preferred to cluster a column with other columns that were close to it and with the more extensive test, so did the rows. There was therefore a type of preferred sequential matching that took place, or the energy value for one time unit was typically closer to the immediately preceding or subsequent time units. To study this further, each mini-cluster was analysed across the row and column indexes, to measure the variability size from one index to the next. For example, if rows or columns 1, 4, 8, 10 were clustered together, then this would produce a variability of 3+4+2 = 9 / 3 = 3. This variability value was measured for the 4 companies separately and is shown in Table 3.

	10% Coverage			50% Coverage		
Company	**Row Var**	**Col Var**	**Av Val**	**Row Var**	**Col Var**	**Av Val**
50Hertz	19.1	5.3	0.697	3.5	4.9	0.250
Amprion	28.8	3.7	0.740	4.4	3.5	0.267
TennetTSO	24.7	8.8	0.711	3.1	4.8	0.243
TransnetBW	37.3	4.4	0.616	4.3	4.7 (14)	0.237

Table 3. Variances for rows and features, for the Germany Wind Power stations.

The results for 50% coverage are with an adjustment for the TransnetBW dataset. If all columns were included, then column 2 would be clustered with columns much further on and this would lead to an anomaly in the sequence variability. If that single column index was ignored in the calculation, then the variances were all much closer. The differences therefore are quite small, but it looks like TransnetBW has the most variability across both rows (seasonal weather) and columns (daily weather). If the anomaly is included however, then the difference is clear and it is still a feature in the TransnetBW dataset that is not in the other ones. So there may be a question about a mountainous terrain affecting the performance there and at what energy range it occurs the most. The other 3 companies appear to be a lot more similar. Amprion performs the best and also has the best day (column) variance values.

6 Conclusions

This paper suggests an architecture that includes both discrete and centralising elements and therefore has some similarities with neural network architectures. A stochastic and discrete layer clusters randomly selected subspaces of the data into mini-clusters, not using gradient descent as in neural networks, but using a Euclidean distance linear classifier, as in [3] or the random networks [18]. Then, an aggregating layer combines the discrete results, rather like a deep learning pooling layer [5]. The algorithm can be trained to recognise similarities across data rows, or data columns (features), in a self-similar way. An optimising feature means that the algorithm prefers to cluster similar rows with larger values first. One might think about an energy surface with peaks and troughs, for example, but the surface is being traversed in many different places at the same time. Overtraining might then be recognised when the localised peak distributions start to merge with each other, which can happen when more lower-valued rows are linked with the higher-valued ones, through the continued aggregation of mini-clusters from a random ordering. In this paper, the architecture is used to cluster and make predictions over energy data, using a stochastic Hyper-Grid [9] and a Frequency Grid [8]. The discrete, localised optimisations in the hyper-grid match dataset rows that are more similar, using a distance measurement, but is also able to discriminate and keep only the row sets that will optimise for some overall total. This method is also described in [3], in terms of bagging and random subspaces, as being a possibility to protect from adversarial attack, by keeping some of the data always hidden.

The case study topic relates to the IDEAS Smart Home Energy Project, where a client-side Artificial Intelligence component can predict energy consumption for appliances. The proposed data model for that is essentially a look-up table of the key energy bands that each appliance would use. Each band represents a level of consumption by the appliance. This table can replace disaggregation from more complicated methods constructed from probability theory, for example. Results show that the table can accurately disaggregate a single source to a set of appliances, because each appliance has quite a unique energy footprint. As part of predicting energy consumption, the model could possibly reduce costs by 50% and more than that if the proposed schedules are also included [12]. The hyper-grid has been changed to consider rows as single units, making it more tractable. A second case study considers wind power patterns, where the grid optimises over the dataset columns that represent a time-series, showing some level of sequential consistency over this feature analysis.

Acknowledgement

This work is supported by the project of "Novel Building Integration Designs for Increased Efficiencies in Advanced Climatically Tuneable Renewable Energy Systems (IDEAS)" (Grant ID: 815271), which is funded by the EU Horizon 2020 programme.

References

[1] Baranski, M. and Voss, J. (2004). Genetic algorithm for pattern detection in NIALM systems. In 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583) (Vol. 4, pp. 3462-3468). IEEE.

[2] Beckel, C., Kleiminger, W., Cicchetti, R., Staake, T. and Santini, S. (2014). The ECO data set and the performance of non-intrusive load monitoring algorithms. In Proceedings of the 1st ACM conference on embedded systems for energy-efficient buildings, pp. 80-89.

[3] Biggio, B., Fumera, G. and Roli, F. (2010). Multiple classifier systems for robust classifier design in adversarial environments, Int. J. Mach. Learn. & Cyber., Vol. 1, pp. 27 – 41. DOI 10.1007/s13042-010-0007-7.

[4] Candanedo., L.M., Feldheim, V. and Deramaix, D. (2017). Data driven prediction models of energy use of appliances in alow-energy house, Energy and Buildings, Vol. 140, pp. 81 – 97. http://dx.doi.org/10.1016/j.enbuild.2017.01.083.

[5] Chen, Y., Sun, X. and Jin, Y., 2019. Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation. IEEE transactions on neural networks and learning systems, 31(10), pp.4229-4238.

[6] ECO Dataset. (2022). http://vs.inf.ethz.ch/res/show.html?what=eco-data.

[7] Efron, B. and Tibshirani, R. (1993). An Introduction to the Bootstrap. Boca Raton, FL: Chapman & Hall/CRC.

[8] Greer, K. (2019). New Ideas for Brain Modelling 3, Cognitive Systems Research, Vol. 55, pp. 1-13, Elsevier. DOI: https://doi.org/10.1016/j.cogsys.2018.12.016.

[9] Greer, K. (2012). A Stochastic Hyper-Heuristic for Matching Partial Information, Advances in Artificial Intelligence, Vol. 2012, Article ID 790485, 8 pages. doi:10.1155/2012/790485, Hindawi.

[10] Holland, J.H. (1984). Genetic algorithms and adaptation. In Adaptive Control of Ill-Defined Systems (pp. 317-333). Springer, Boston, MA.

[11] IDEAS. (2022). Novel building Integration Designs for increased Efficiencies in Advanced climatically tunable renewable energy Systems, https://www.horizon2020ideas.eu/.

[12] IDEAS, Deliverable 4.4. (2021). Smart Mobile App Development, https://www.horizon2020ideas.eu/publications-papers-downloads-videos-about-ideas/.

[13] Jiang, X., Xiao, C. and Sun, J. (2019) Household Energy Demand Management Strategy Based on Operating Power by Genetic Algorithm, IEEE Access, Vol. 7, pp. 96414 - 96423. DOI 10.1109/ACCESS.2019.2928374.

[14] Kephart, J.O. and Chess, D.M. (2003). The vision of autonomic computing. Computer, 36(1), pp.41-50.

 doi:10.20944/preprints202201.0313.v1

[15] Kim, H., Marwah, M., Arlitt, M., Lyon, G. and Han, J. (2011). Unsupervised disaggregation of low frequency power measurements. In Proceedings of the 2011 SIAM international conference on data mining (pp. 747-758). Society for Industrial and Applied Mathematics.

[16] Kolter, J.Z. and Jaakkola, T. (2012). Approximate inference in additive factorial hmms with application to energy disaggregation. In Artificial intelligence and statistics (pp. 1472-1482). PMLR.

[17] Li, K. and Malik, J. (2018). Implicit Maximum Likelihood Estimation, arXiv:1809.09087v2 [cs.LG] 22 Oct 2018.

[18] Liang, X., Javid, A.M., Skoglund, M. and Chatterjee, S. (2022). Decentralized learning of randomization-based neural networks with centralized equivalence, Applied Soft Computing, Vol. 115, ISSN 1568-4946, https://doi.org/10.1016/j.asoc.2021.108030.

[19] Licas. https://licas.sourceforge.io/.

[20] Noorollahi, Y., Aligholian, A. and Golshanfard, A. (2019). Stochastic energy modeling with consideration of electrical vehicles and renewable energy resources - A review, Journal of Energy Management and Technology (JEMT) Vol. 4, Issue 1, pp. 13 - 25.

[21] Wind Power Generation Data. (2022). https://www.kaggle.com/jorgesandoval/wind-power-generation.