Data Mining and Knowledge Discovery (2019) 33:413–445
https://doi.org/10.1007/s10618-018-0598-2

# Interpretable multiple data streams clustering with clipped streams representation for the improvement of electricity consumption forecasting

**Peter Laurinec**[1] · **Mária Lucká**[1]

Received: 26 November 2017 / Accepted: 29 October 2018 / Published online: 16 November 2018


**Abstract**
This paper presents a new interpretable approach for multiple data streams clustering in a smart grid used for the improvement of forecasting accuracy of aggregated electricity consumption and grid analysis named *ClipStream*. Consumers time series streams are compressed and represented by interpretable features extracted from the clipped representation. The proposed representation has low computational complexity and is incremental in the sense of the windowing method. From the extracted features, outlier consumers can be simply and quickly detected. The clustering phase consists of three parts: clustering non-outlier representations, the aggregation of consumption within clusters, and unsupervised change detection procedure on aggregated time series streams windows. *ClipStream* behaviour and its forecasting accuracy improvement were evaluated on four different real datasets containing variable patterns of electricity consumption. The clustering accuracy with the proposed feature extraction method from the clipped representation was evaluated on 85 time series datasets from a large public repository. The results of experiments proved the stability of the proposed *ClipStream* in the sense of improving forecasting accuracy and showed the suitability of the proposed representation in many tested applications.

**Keywords** Data streams clustering · Time series representations · Electricity consumption forecasting

Responsible editor: Jesse Davis, Elisa Fromont, Derek Greene, Bjorn Bringmann.

✉ Peter Laurinec
peter.laurinec@stuba.sk

Mária Lucká
maria.lucka@stuba.sk

[1] Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, Ilkovičova 2, 842 16 Bratislava, Slovak Republic

## 1 Introduction

The vast amount of fast incoming data from various types of meters are stored and can be used for further mining to bring actionable insights. An example of such a scenario is a smart grid ecosystem that can consist of millions of smart meters. Smart meters measure and send information from consumers or producers (together prosumers) about their electricity consumption, production, lagging reactive and leading reactive power and other similar data. This information can be used to make important decisions for regularisation of production, forecasting future states or for helping recommend to consumers how to set a reserved capacity of consumption (bill settings). Monitoring the grid can help for the purpose of preventing blackouts and other factors damaging the electricity network. Real-time results are in demand for these kinds of situations. As data from smart meters are received continuously, they can be referred to as data streams (i.e., time series streams). Therefore this forces us to develop high-performance techniques for data stream mining for real-time analysis and decision making in the smart grid.

For this purpose, we proposed an interpretable multiple data streams clustering approach for monitoring and analysing smart grid, creating consumer profiles, detecting important changes in the grid and improving forecasting accuracy of aggregated electricity consumption by creating more predictable groups of consumers. To achieve these goals, we are using newly-proposed, fast and interpretable techniques that support these actions. Also, other types of sensors (meters) produce a large amount of data that have a time series structure and this forces us to develop more general methods for data mining. This important factor is also analysed and evaluated in our work.

This paper is structured as follows: Sect. 2 contains related works. Section 3 presents a description of our proposed approach and forecasting methods. Section 4 presents the description and the evaluation of performed experiments alongside with the benchmark methods and datasets used, and the paper concludes with Sects. 5 and 6, where we discuss achieved results.

### 1.1 Contributions

We have proposed *ClipStream*, an attribute multiple data stream clustering approach for extraction of interpretable insights from consumers data and for improving forecasting accuracy of the aggregated times series of electricity consumption.

Contributions of our work are as follows: 1) a newly proposed feature extraction method from clipped representation of data streams (*FeaClip*), which is fast and incremental in nature; 2) a proposal for an interpretable and automatic outlier detection method directly from proposed representation; and 3) a newly proposed unsupervised distribution change detection method for decreasing computational consumption of *ClipStream* clustering.

## 2 Related work

Data stream clustering is a rapidly evolving part of data mining. Silva et al. (2013) created an extensive survey dealing with this phenomenon, in which types of data stream clustering techniques and corresponding challenges were discussed. Most of the attention in literature so far has been placed on object-based data stream clustering that focuses on the one data stream clustering. Representative methods in this category are for example ClusStream (Aggarwal et al. 2003), D-Stream (Chen and Tu 2007), DBSTREAM (Hahsler and Bolaños 2016), MuDi (Amini et al. 2016), and Str-FSFDP (Chen and He 2016). However, we are focused on attribute-based data stream clustering that clusters multiple streams (consumers in our case) through their extracted attributes. An attribute-based data stream clustering method should ideally take into account restrictions as similar to any data stream clustering algorithm as follows: a) performing fast and incremental processing of data objects; b) rapidly adapt to changing dynamics of the data (merging and removing of clusters); c) scale to the number of objects that are continuously arriving; d) rapidly detect the presence of outliers and act accordingly (Silva et al. 2013). The authors of the survey also stated that change detection methods are missing in the process of the most proposed data stream clustering methods. The multiple data streams (or time series streams) clustering problem was tackled in the works of Dai et al. (2006) (COD), Beringer and Hüllermeier (2007) (FCM-DS), Chen (2009) (CORREL-cluster), Rodrigues et al. (2008) (ODAC), Chen et al. (2012) (IDEStream), Pereira and de Mello (2014) (TS-stream), and Khan et al. (2016) (SPE-cluster). These works focus on the correlation-based comparison of streams, the clustering of Fourier or wavelet coefficients, and feature extraction from generative functions of time series streams or density-based ensemble clustering. In our work, we are focused on more interpretable and high-performance techniques for purposes such as improving forecasting accuracy, consumers analysis, and smart grid monitoring. Change and outlier detection methods for multiple data streams were used in the work of Appice et al. (2014). They use temporal and spatial correlations alongside a supervised technique based on exponential smoothing forecasting errors to detect anomalous sensors or changes. Pravilovic et al. (2017) also use spatial information for multiple time series forecasting. They use the multivariate VAR model with principal components as independent variables for forecasting. Both works use many useful data stream mining techniques, however, we are not considering spatial information to be used, or supervised (or semi-supervised) detection of outliers or changes. The biggest impact on the amount of electricity consumption has the type of consumer, and choice of date and time of a measurement. The amount of electricity consumption is affected mainly by multiple seasonalities-daily, weekly and yearly.

As analysis of time series streams of electricity consumption is our goal, the use of time series data mining methods is highly recommended. Methods of representations of time series are used for dimensionality reduction and for emphasising characteristics of data (Esling and Agon 2012). Representation methods can be divided into four groups: nondata adaptive, data adaptive, model-based and data dictated (Ratanamahatana et al. 2005). In nonadaptive representations, the parameters of transformation remain the same for all time series, irrespective of their nature. The most used nondata

adaptive methods are PAA (Keogh and Pazzani 2000), DFT (Faloutsos et al. 1994), or DWT (Chan and Fu 1999). In data adaptive representations, the parameters of transformation vary depending on the available data. The representatives are PLA (Keogh and Pazzani 1998), APCA (Keogh et al. 2001), and SAX (Lin et al. 2003). An approach to the model-based representation relies on the assumption that the observed time series was created from some basic model such as ARIMA (Corduas and Piccolo 2008) or linear regression (Laurinec and Lucká 2016). In data dictated approaches, the compression ratio is defined automatically based on a raw time series such as a clipped representation (Aghabozorgi et al. 2015).

Aghabozorgi et al. (2015) highlighted that density-based clustering algorithms for time series are not applicable because of their rather high complexity. It also implies that many density-based data stream clustering algorithms are not appropriate in our case. Paparrizos and Gravano (2015) proposed a new centroid-based clustering method for time series called k-Shape. As its distance measure, k-Shape uses a normalised version of the cross-correlation measure in order to consider the shapes of time series while comparing them. The k-Shape algorithm was already used in two works for improving electricity consumption forecasting (Yang et al. 2017; Jarábek et al. 2017), therefore it is an appropriate benchmark method.

Smart meter data clustering for improving forecasting accuracy with the ODAC method (Rodrigues et al. 2008) and the neural network was used in the work of Gama and Rodrigues (2007). Our previous works Laurinec and Lucká (2016); Laurinec et al. (2016); Laurinec and Lucká (2017, 2018b) focused on representations of time series and various forecasting methods to improve forecasting accuracy with the combination of K-means clustering. We contested that model-based representations are best for extracting consumers patterns, but they are not appropriate for data streams due to their high computational complexity. However, it has been proved that clustering various types of representations gives more accurate results than clustering the original time series of the consumption.

## 3 ClipStream: multiple data streams clustering

The *ClipStream* consists of two main steps: data abstraction and an offline phase as can be seen in Fig. 1. The data abstraction step covers the fast and incremental computation method of creating feature vectors from stream windows named *FeaClip* and automatic detection of outliers directly from *FeaClip*. The offline phase consists of clustering of representations of data streams, the aggregation of clustered time series and the change detection process. After the *ClipStream* process, forecasting methods are applied to created clustered and aggregated time series of electricity consumption to benefit from *ClipStream* clustering.

The list of used symbols in the paper can be found in Table 1.

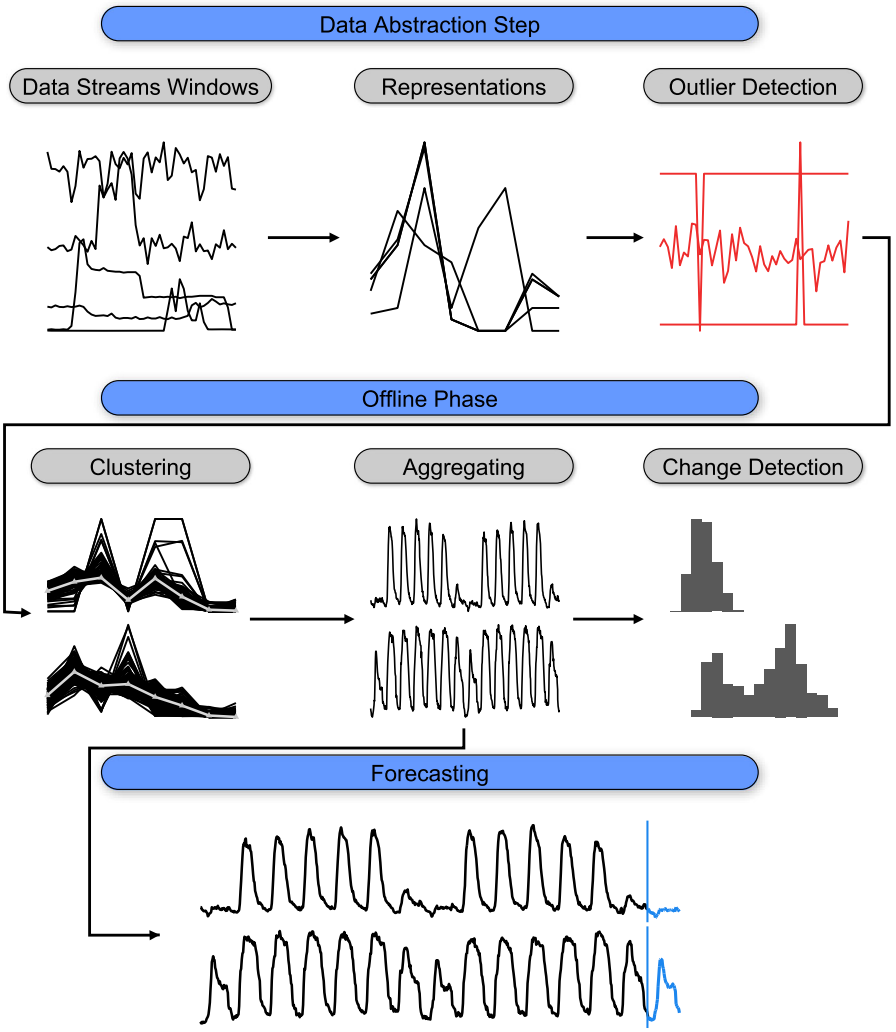


**Fig. 1** Steps of proposed *ClipStream* approach for multiple data streams clustering of smart meter data

### 3.1 Representation of a data stream

We proposed a new feature extraction approach from a clipped representation (class of data dictated representation methods) of data stream named *FeaClip* for electricity consumption time series that is inspired by the work of Bagnall et al. (2006). Two initial simple transformations of the original data stream were used from this work before our proposed feature extraction procedure is then used to extract a final representation of a data stream.

Firstly, let us formally define a data stream (Silva et al. 2013), a short and long window from the data stream and a representation of the data stream.

**Table 1** Table of used symbols

| | |
|---|---|
| $\mathbf{s}$ | Data stream |
| x | Object, observation |
| $T$ | Length of time series |
| $\mathbf{w}^l$ | Long window |
| $\mathbf{w}^s$ | Short window |
| $n_{w^s}$ | Length of short window |
| $u$ | Number of short windows |
| $\mathbf{repr}^l$ | Representation of long window |
| $\mathbf{repr}^s$ | Representation of short window |
| $\mu$ | Mean |
| $\sigma$ | Standard deviation |
| $K$ | Number of clusters |
| $\lambda$ | Outlier detection threshold |
| $\alpha$ | Change detection threshold |

**Definition 1** Data stream $\mathbf{s}$ is a sequence of objects $\mathbf{s} = x_1, x_2, \ldots, x_T$, or $\mathbf{s} = \{x_t\}_{t=1}^{T}$, which is potentially unbounded ($T \rightarrow \infty$).

**Definition 2** Short window $\mathbf{w}^s$ is a subsequence of data stream $\mathbf{s}$ of length $n_{w^s}$, $\{w_t^s\}_{t=1}^{n_{w^s}}$. Long window $\mathbf{w}^l$ consists of last $u$ non-overlapping successive short windows $\mathbf{w}^s$ from a data stream $\mathbf{s}$, so it has length of $u * n_{w_i^s}$, where $i = 1, \ldots, u$.

**Definition 3** Let $\mathbf{w}^s$ be a subsequence of data stream $\mathbf{s}$ of length $n_{w^s}$, then representation of $\mathbf{w}^s$ is a model $\mathbf{repr}^s$ with reduced dimensionality $p$ ($p < n_{w^s}$) such that $\mathbf{repr}^s$ approximates closely $\mathbf{w}^s$.

We use a short window $\mathbf{w}^s$ of the data stream $\mathbf{s}$, then

$$\hat{w}_t^s = \begin{cases} 1 & \text{if} \quad w_t^s > \mu \\ 0 & \text{otherwise} \end{cases}, \tag{1}$$

$\hat{\mathbf{w}}^s$ is a clipped (bit-level) representation of the original subsequence, where $\mu$ is the average value of $\mathbf{w}^s$ and $t = (1, \ldots, n_{w^s})$. The compression method Run Length Encoding (RLE) is applied to the clipped representation $\hat{\mathbf{w}}^s$. A run is a continuous sequence of ones respectively zeros. The number of ones respectively zeros in a run we call the run lengths. From the run lengths counted by RLE, eight simple interpretable features are extracted to form final representation $\mathbf{repr}^s$ from short window $\mathbf{w}^s$ and is defined as

$$\begin{aligned}\mathbf{repr}^s = \{&max_1 = \text{max. from run lengths of ones},\\ &sum_1 = \text{sum of run lengths of ones},\\ &max_0 = \text{max. from run lengths of zeros},\\ &crossings = \text{length of RLE encoding} - 1,\\ &f_0 = \text{number of first zeros},\\ &l_0 = \text{number of last zeros},\\ &f_1 = \text{number of first ones},\\ &l_1 = \text{number of last ones}, \}.\end{aligned} \tag{2}$$

If the first run length is zero then it implies that $f_1$ will be zero and if the last run length is zero then it implies that $l_1$ will be zero and vice versa.

For example, consider a clipped (bit-level) representation of $\mathbf{w}^s$ as

$$11111100001100000111111100000,$$

then the corresponding RLE values are 6#1, 4#0, 2#1, 5#0, 6#1, 5#0 and the final representation is $\mathbf{repr}^s = \{max_1 = 6,\ sum_1 = 14,\ max_0 = 5,\ crossings = 5,\ f_0 = 0,\ l_0 = 5,\ f_1 = 6,\ l_1 = 0\}$.

For the creation of the final *FeaClip* representation $\mathbf{repr}^l$ of the long window $\mathbf{w}^l$ of a data stream, a windowing approach is used. Objects are added to a short window $\mathbf{w}^s$ until it's length is $n_{w^s}$ or the time limit is exceeded. Next, $\mathbf{repr}^s$ representation is computed. Then $u - 1$ new short windows are created and corresponding representations are computed until a long window $\mathbf{w}^l$ of length $u * n_{w_i^s}$, where $i = 1, \ldots, u$, is processed. So, the final representation $\mathbf{repr}^l$ is an union of short representations $\mathbf{repr}^s$ having length $u * 8$. If another new window $\mathbf{w}^s$ is filled and the corresponding representation $\mathbf{repr}^s$ is computed, first 8 features from $\mathbf{repr}^l$ are removed then new 8 from $\mathbf{repr}^s$ are bound to the end of $\mathbf{repr}^l$. The visualisation of the created *FeaClip* representation, $\mathbf{repr}^l$, of a randomly picked consumer, where long window $|\mathbf{w}^l| = 21$ days and short window $|\mathbf{w}^s| = 1$ day, is shown in Fig. 2. It illustrates that the original dimension was reduced by $(21 * 48)/(21 * 8) = 6$ multiple times. Short data stream windows resp. representations ($\mathbf{w}^s$ resp. $\mathbf{repr}^s$) are bordered by grey dashed lines.

The proposed *FeaClip* representation captures two types of behaviour of the data stream. The first four features ($max_1$, $sum_1$, $max_0$ and $crossings$) capture the global statistics of a data stream, including the maximal run length above and under average, the sum of all counts of observations above average and the number of crossing points around average. From these features, outlier consumers can be derived simply and interpretably (explained in Sect. 3.2). The last four features ($f_0$, $l_0$, $f_1$ and $l_1$) capture the local behaviour of the data stream and serve for more precise separation, so can be used for distinguishing how the data window starts and ends. One of the many situations where these local features help can be seen in Fig. 3 where *FeaClip* representations, $\mathbf{repr}^s$, for randomly picked consumer (short window $\mathbf{w}^s$) and its reversed values are shown. Representations differ from each other only in features $f_0$ and $l_0$, because they have a different count of observations under average at the beginning and at the end of the time series. These four local features, if we imagine electricity consumption time

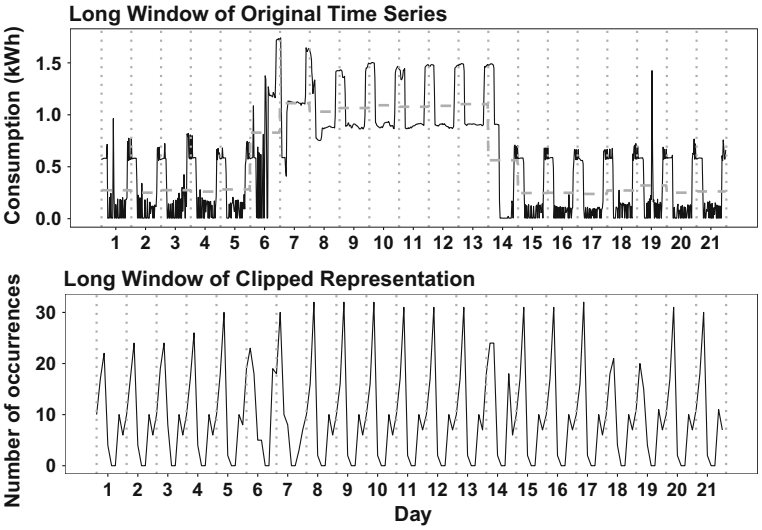


**Fig. 2** The original time series (the long window of data stream) and its corresponding long window of *FeaClip* representation of a randomly picked consumer. The horizontal dashed line represents the average values of consumption during the day (i.e. short window) and the vertical dotted lines limit the short windows of the length of one day

series, explain how long non-consumption of electricity lasts at the start or the end of a day, or vice versa - consumption of electricity. It is important to use them because these four local features are dependent on time since the first four global features are not.

Our proposed *FeaClip* representation has the following advantages: a) it emphasises the main characteristics of data; b) all features are simply interpretable; c) it reduces dimension; d) there is no need for normalisation by z-score or min-max method; e) short windows can have a different number of observations; f) it is incremental (windowing) and fast to compute; g) and clipping and features can be computed in parallel.

### 3.2 Outlier detection

From computed *FeaClip* representations of all available data streams windows, outliers (consumers) can be simply and automatically detected by the usage of domain knowledge. Suspicious consumers can be those in which periodically, consumption jumps to very high values. The reverse situation happens when consumption is holding some value and suddenly jumps rapidly down (due to mechanical failure or blackout). These two situations can be simply detected by analysing the feature $sum_1$ from *FeaClip* because it details by how much observations are over or above average, when $sum_1$ has maximal or minimal values. Another anomalous situation is when a large number of crossings around the average are observed. It implies that consumer behaviour is vibrant, unstable and unpredictable. This can be detected only by analysing the feature $crossings$.

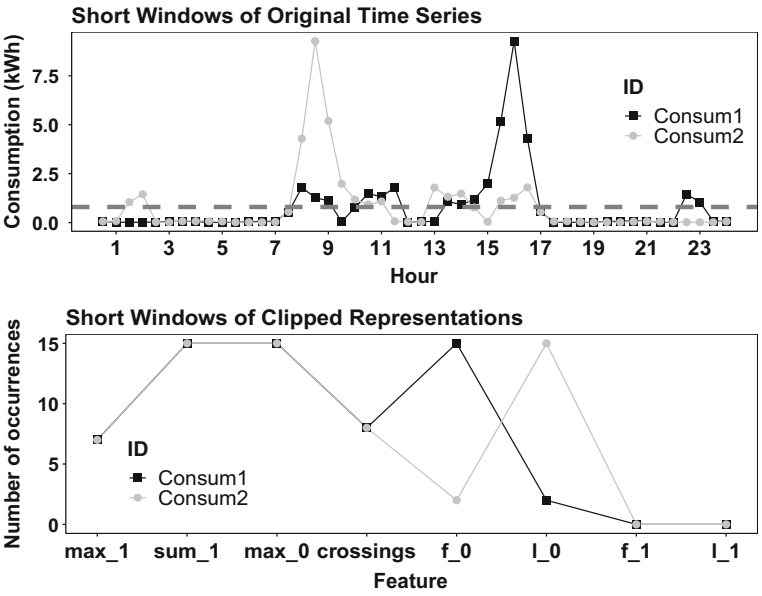


**Fig. 3** The short windows of *FeaClip* representations of a randomly picked consumer and its reverse values of the length of one day. The dashed line represents the average value of consumption during the day (i.e. short window)

To automatise outlier detection, firstly, average values of features $sum_1$ and $crossings$ are computed for every data stream window and corresponding actual *FeaClip* representation $\mathbf{repr}^l$. Then, from average values of the $sum_1$, upper and lower quartiles, and IQR (interquartile range) are computed and for averages of the feature $crossings$, the upper quartile and IQR are computed. The data stream window is labelled as an outlier if it satisfies at least one of these three conditions: average value of $sum_1$ is higher than $Q_3^{sum_1} + \lambda * IQR^{sum_1}$, the average value of $sum_1$ is lower than $Q_1^{sum_1} - \lambda * IQR^{sum_1}$ or the average value of $crossings$ is higher than $Q_3^{crossings} + \lambda * IQR^{crossings}$, where $\lambda$ is the threshold coefficient. It follows standard box-and-whisker diagram (plot) methodology of finding outlier observations in a sample, when $\lambda = 1.5$ (McGill et al. 1978). However, in our scenario the $\lambda$ can vary around the value 1.5.

The distribution of outlier and normal average values of *FeaClip* representations of consumers are shown in Fig. 4. Long tails of outlier distribution can be seen in almost every feature, which proves the suitability of our approach even though outliers are detected based only on two features: $sum_1$ and $crossings$.

Examples of extracted outlier consumers time series are shown in Fig. 5.

The position of 49 outlier consumers from 989 in a PCA and t-SNE 2D space based on Fig. 4 is shown in Fig. 6. We can see that extracted outliers mostly lie on a border of greater clusters or as standalone points in a space.

Outliers are not removed from the whole clustering process because they are needed for forecasting of aggregated consumption. They are held in memory and after the

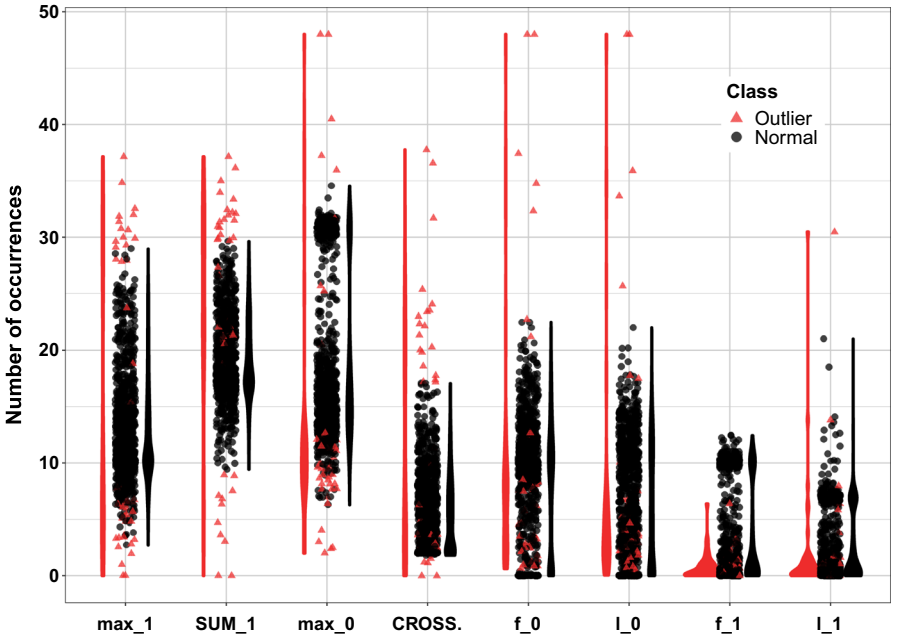


**Fig. 4** The distribution of outlier and normal *FeaClip* representations of consumers time series according to the method described in Sect. 3.2. Outlier consumers were determined based on abnormal values of features *SUM_1* and *CROSS.*. CROSS. refers to the number of crossing points

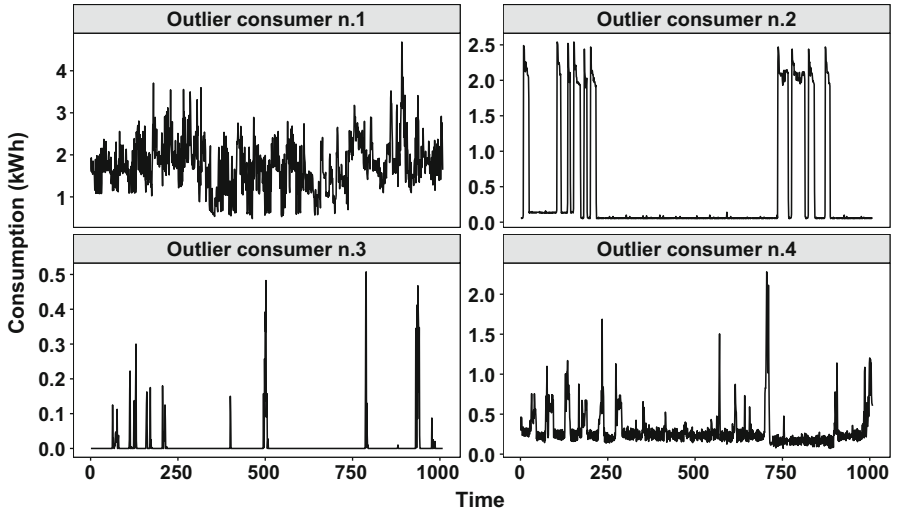


**Fig. 5** Four extracted outlier consumers time series of length three weeks based on Fig. 4

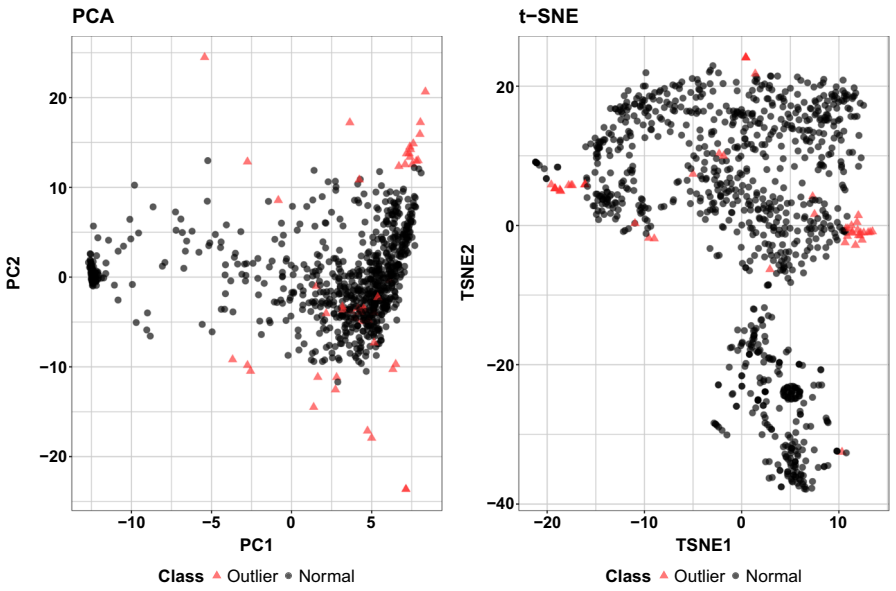


**Fig. 6** The position of 49 outlier consumers from 989 in a PCA and t-SNE 2D space based on Fig. 4

creation of clusters, outliers are assigned to the nearest ones. This process accelerates the computation of the clustering algorithm because the number of observations is reduced, and the stability of clustering results is also improved. Another advantage is that further analysis and recommendations for power usage for outlier consumers are possible after detection.

### 3.3 Clustering data streams representations

After the data abstraction step where representations of multiple data streams have been created, the offline phase follows, to create a final clustering. Only filtered (non-outlier) long windows of data streams representations are clustered. For filtered *FeaClip* representations clustering, K-medoids method with Partition Around Medoids (PAM) algorithm is used (Kaufman and Rousseeuw 2009). The K-medoids method minimises distances between objects in clusters and corresponding medoids. The big advantage here is that any distance measure can be used. We used Euclidean distance measure because we have to use feature wise distance comparison for *FeaClip* representations. K-medoids has more stable and accurate results on noisy datasets (i.e. with large amount of outliers) (Arora et al. 2016; Manjoro et al. 2016), therefore K-medoids is used.

To capture the dynamic and evolving nature of data streams, the number of clusters also have to be determined dynamically. Therefore an optimal number of clusters was determined by the internal validation measure Davies–Bouldin index (Davies and Bouldin 1979). In the first iteration of data streams clustering, the number of possible clusters is determined in the range of $K_{min} - K_{max}$. $K$ with a minimal value of

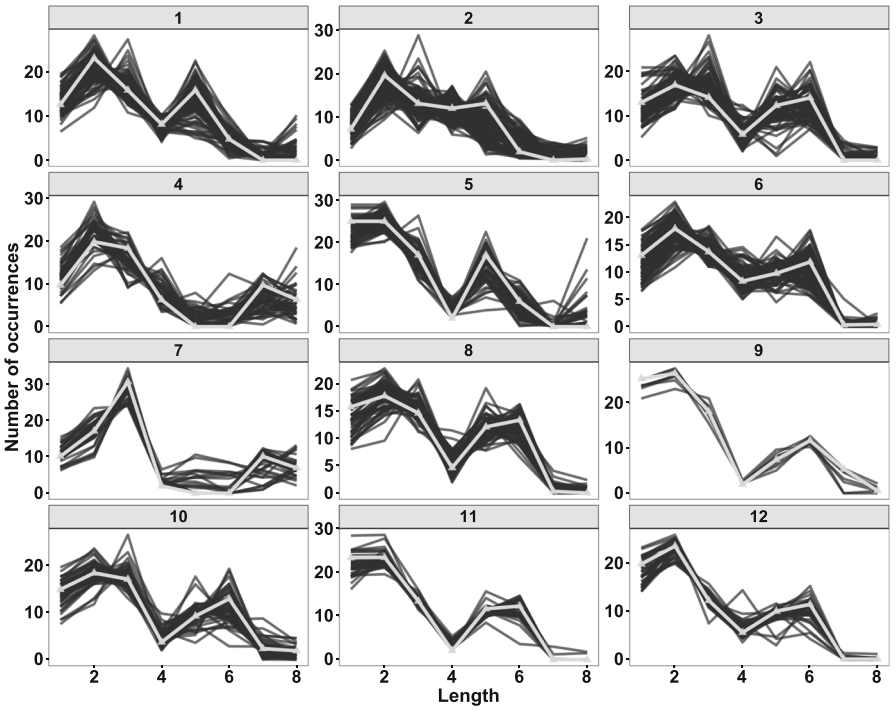


**Fig. 7** 12 clusters of average daily *FeaClip* representations of consumers time series created by K-medoids. Medoids are drawn by a light grey line

the Davies–Bouldin index chosen. To accelerate further iterations (windows) of data streams clustering, the optimal number of clusters is chosen among $\langle K-2, K+2\rangle$, where $K$ is the number of clusters in the previous data window step.

By K-medoids clustering, the labels of clustered objects and $K$ medoids are extracted. This information is then used to quickly assign the outliers to the nearest medoid with relation to the Euclidean distance. After this computation, the final clustering labels of all *FeaClip* representations of data streams are done.

Created clusters of consumers, without outliers, can be seen in Fig. 7. Average daily *FeaClip* representations are shown, even though whole representations $\mathbf{repr}^l$ of length $8 * 21$ are clustered, because the entire clustered representations would be too long to visualise.

### 3.4 Aggregating clusters

Our objective is, by the clustering of consumers, to create more predictable (forecastable) groups of aggregated time series for forecasting (e.g. summated electricity consumption of a region). Therefore, the original time series data streams stored in long windows are summated by the clustering created in the previous step, while preserving its frequency (granularity). Final clustered and aggregated time series are then

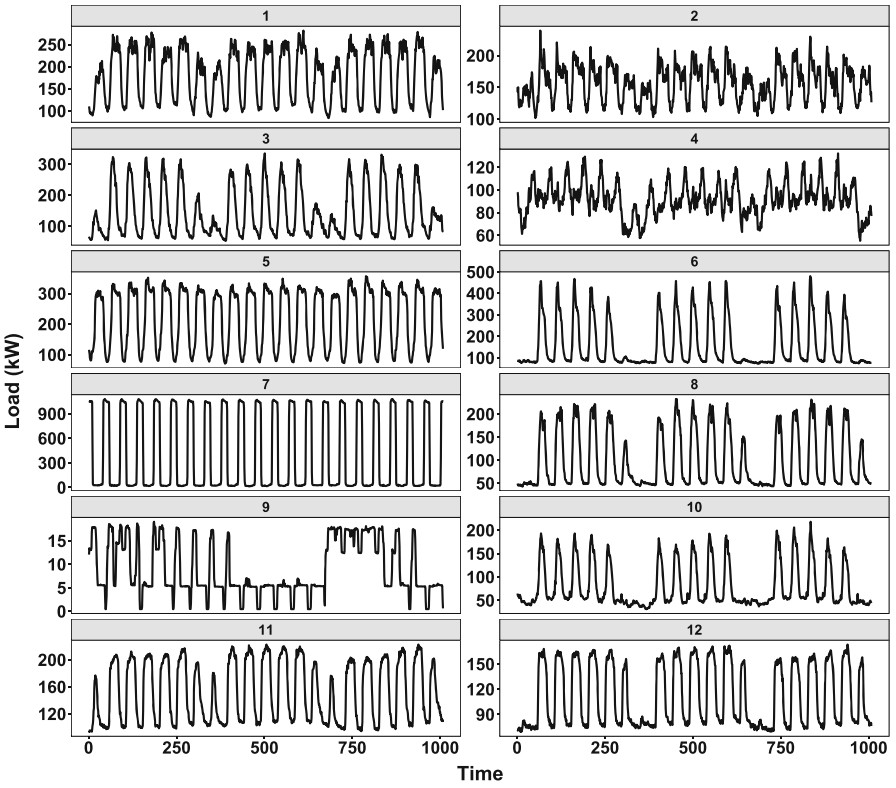


**Fig. 8** Final aggregated time series within clusters based on Fig. 7 which will give input to forecasting methods

ready to input to forecasting methods. The 12 aggregated time series within clustering created by methods described in Sect. 3.3 are shown in Fig. 8.

If we would like to forecast, for example, the individual consumption of a consumer, the clustering results can be also used to create new robust training sets (Laurinec and Lucká 2018a). Mean or median can be used in this scenario for an aggregation of time series within clusters.

### 3.5 Change detection

In order to perform the data streams clustering process only when necessary, so only when data streams evolve and change of distribution occurs, the change detection method is proposed. This process highly decreases computational load because the offline phase of data streams clustering (and our *ClipStream* approach) is the most time consuming (complex) part.

We are interested in the testing of distribution comparisons from only normalised values because classical trend decreasing and increasing (i.e., incremental concept drift) is handled by our implemented forecasting methods. Our goal is the change

detection of the Empirical Distribution Function (EDF) of the normalised data stream windows. Unsupervised statistical change detection method using k-sample Anderson-Darling test (Scholz and Stephens 1987) for equality of distributions was chosen in order to detect changes in the aggregated data stream window. The k-sample Anderson-Darling test is based on the quadratic differences of EDF values from k samples. The Anderson-Darling test for distribution testing is a widely used method, in the work of Razali et al. (2011), statistical tests power comparisons were evaluated for normality testing, where the Anderson-Darling test had better results than other known methods such as the Kolmogorov–Smirnov test and the Cramer-von Mises test. The unsupervised statistical change detection method was chosen because interventions to the machine learning method (model) can be applied immediately in comparison to error-based supervised methods, and except for significance level ($\alpha$), the method is parameter free, so there is no need for thresholds settings for prediction errors.

Every aggregated time series window obtained from the previous procedure (Sect. 3.4) is tested for change detection. Time series windows are cut to lengths of one week, so, for example, if the long window has a length of three weeks, we have three subseries for testing. This is due to a double-seasonal pattern of electricity consumption time series (daily and weekly seasonality). By this, we are testing if the weekly pattern is same in each week window (subseries), so it is simple to forecast. Before applying the k-sample Anderson-Darling test, all subsets of time series are normalised by the min–max method, in order to have the same scale. Min-max normalisation is defined as follows:

$$x' = \frac{x - \min(\mathbf{x})}{\max(\mathbf{x}) - \min(\mathbf{x})}, \tag{3}$$

where $x$ is the original observation, $\mathbf{x}$ is window of observations and $x'$ is the normalised observation.

Let $x_{ij}$ be the $j$-th observation in the $i$-th sample ($j = 1, \ldots, t_i; i = 1, \ldots, k$). All observations are independent. Let $T = t_1 + t_2 + \cdots + t_k$ be a size of all samples (the size of the long window). The k-sample Anderson-Darling test statistic is then defined as

$$A^2_{kT} = \frac{1}{T} \sum_{i=1}^{k} \frac{1}{t_i} \sum_{j=1}^{T-1} \frac{(T m_{ij} - j t_i)^2}{j(T-j)}, \tag{4}$$

where $m_{ij}$ is the number of observations in the $i$-th sample that are not greater than $z_j$, where $z_1 < \cdots < z_T$ is the pooled ordered sample (long window).

Single change detection is declared if the $p$ value of the EDF test is less than significance level $\alpha$. Multiple data streams clustering is then updated only if at least one of these two conditions are fulfilled:

1. Number of change detections are more than half of the number of clustered time series $K$.
2. Number of change detections is greater than in the previous sliding window step.

In other words, *ClipStream* update is performed only if there are more than $K/2$ EDF change detections or the number of change detections increases.

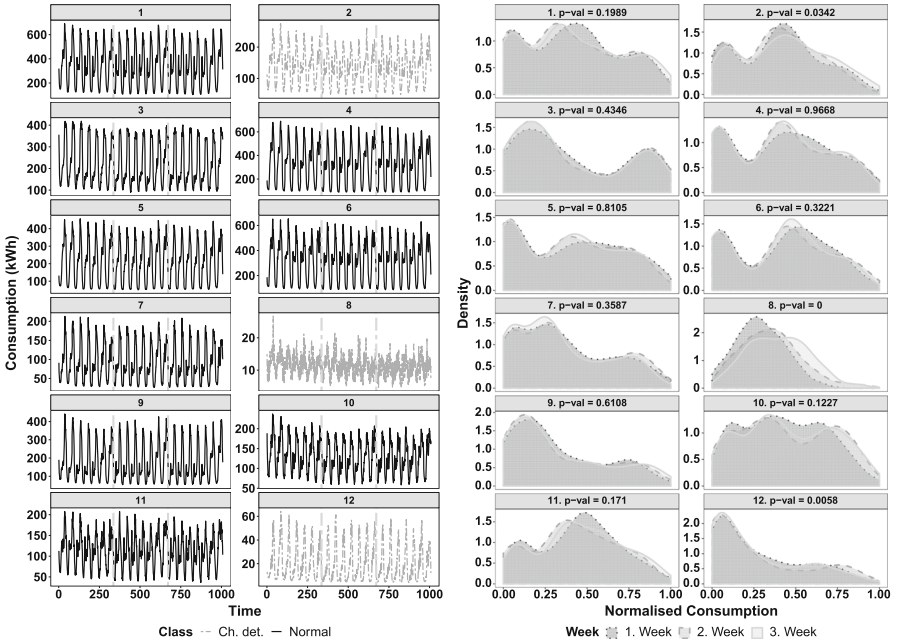


**Fig. 9** The EDF change detection procedure on aggregated consumers time series windows of length three weeks. Green dashed lines border subseries of a length of one week. Titles of stripes above density graphs contain the $p$ values of the Anderson-Darling test. Change detection is declared here if the $p$ value is lower than 0.05

The change detection procedure on 12 aggregated data stream windows is shown in Fig. 9. We can see three aggregated long data stream windows in which EDF changes were detected by a 3-sample Anderson-Darling test.

### 3.6 Forecasting methods

Four different forecasting methods, of which two have two different variations, were implemented to prove that *ClipStream* clustering significantly improves the forecasting accuracy of electricity consumption. These kind of forecasting methods were chosen as they can benefit from clustering, so they can therefore automatically adapt to various patterns of time series. Forecasting for one day ahead (one short window ahead) is performed, i.e., short-term forecasting.

The accuracy of forecasts was measured by MAPE (Mean Absolute Percentage Error). MAPE is defined as follows:

$$\text{MAPE} = 100 \times \frac{1}{T} \sum_{t=1}^{T} \frac{|x_t - \overline{x}_t|}{x_t}, \tag{5}$$

where $x_t$ is real consumption, $\overline{x}_t$ is forecast consumption and $T$ is a length of the time series.

One time series analysis method and three regression tree-based methods were implemented, and the description of them and the corresponding parameter selection is expanded on below.

**EXP** Triple Exponential Smoothing is a forecasting method applied to a seasonal time series, whereby past observations are not weighted equally, but the weights decrease exponentially with time (Hyndman et al. 2008). In order to adapt to various patterns in electricity consumption data, three different models were fitted each time and the best was picked to produce a forecast. These models comprised a full additive model with a trend component, an additive model with a damping trend and an additive model without a trend. The best model among the three was chosen according to the best values of Akaike information criterion (AIC).

**CART** Recursive partitioning regression tree (Breiman et al. 1984) (Classification and Regression Trees) search over all possible splits by maximising an information measure of node impurity, selecting the covariate showing the best split.

As mentioned in Sect. 3.5, our implemented forecasting methods are adaptable to trend changing. For all regression tree methods (CART, CTREE, and RF), a dependent variable (time series of electricity consumption) was detrended by STL decomposition (Cleveland et al. 1990) in order to improve forecasting accuracy. From the extracted trend part of the time series, future values were forecast by fully automatic ARIMA procedure (Hyndman and Khandakar 2008) and added to the forecast from the regression tree model. Regression tree methods always modelled and forecast an aggregated seasonal and remainder part from the STL decomposition.

As attributes (independent variables) to every regression tree model, double seasonal Fourier terms were created. Fourier signals are perfect for modelling seasonal time series because they consist of periodic trigonometric functions. They are also more appropriate to use in modelling when seasonalities are long (also our scenario), this problem was discussed in work of Livera et al. (2011).

The daily seasonal Fourier term has $ds$ pairs of terms

$$\left(\sin\left(\frac{2\pi\, jt}{48}\right),\ \cos\left(\frac{2\pi\, jt}{48}\right)\right)_{j=1}^{ds} \tag{6}$$

and the weekly seasonal Fourier term has $ws$ pairs of terms

$$\left(\sin\left(\frac{2\pi\, jt}{7}\right),\ \cos\left(\frac{2\pi\, jt}{7}\right)\right)_{j=1}^{ws}, \tag{7}$$

where $t = (1, \ldots, T)$. For the CART method, it was experimentally verified that the best setting of the number of Fourier term pairs are $ds = 2$ and $ws = 4$. Hyperparameters for CART were set as follows: minimum number of observations needed in node to split was set to 2, maximal depth of a tree was 30 and the complexity parameter was 1E-6.

**CTREE** Conditional inference trees (CTREE) is a statistical approach to recursive partitioning, which takes into account the distributional properties of the measurements (Strasser and Weber 1999). CTREE performs multiple test procedures that are applied to determine whether no significant association between any of the covariates and the response can be stated and if the recursion needs to stop.

Two variations of CTREE were tested in experiments which differ in input attributes to the model. One of which was a CTREE.lag variation which uses double seasonal component (part) from STL decomposition with one day lag as the attribute to the model. Additional attributes to the model were seasonal Fourier terms with $ds = 2$ and $ws = 4$ pairs as defined in Eqs. 6 and 7. With the CTREE.dft variation, input attributes were only Fourier terms with $ds = 5$ and $ws = 10$ pairs. Hyperparameters were set as follows for CTREE: the minimal criterion that must be exceeded in order to implement a split was set to 0.95 and the maximal depth of a tree was unbounded.

**RF** Random Forests is an ensemble learning method that constructs a large number of trees and outputs the mean prediction of the individual regression trees (Breiman 2001).

Two variations of the RF similar to CTREE were created. RF.lag has the same attributes as CTREE.lag, but the number of Fourier terms was $ds = 3$ and $ws = 6$ pairs. RF.dft has analogically the same attributes as CTREE.dft, but the number of Fourier terms was $ds = 3$ and $ws = 6$ pairs. Hyperparameters were set as follows for RF: the number of trees was set to 1100, the minimum size of terminal nodes was 3 and the number of variables randomly sampled at each split was 3.

### 3.7 Summary of ClipStream

Detailed steps of the whole *ClipStream* method are as follows:

1. Store multiple data streams in long data stream windows $\mathbf{w}^l$ of predefined length.
2. Compute for every short data stream window $\mathbf{w}^s$ its *FeaClip* representation $\mathbf{repr}^s$.
3. Detect outlier consumers based on *FeaClip* features $sum_1$ and $crossings$.
4. Cluster non-outlier *FeaClip* long windows $\mathbf{repr}^l$ in the range of number of clusters $\langle K_{min}, K_{max} \rangle$ by K-medoids. Choose the best $K$ based on DB-index.
5. Assign outlier consumers to the nearest medoids acquired in the previous step. With this, the full vector of clustering is created.
6. Aggregate (sum) stored time series streams in long data stream windows ($\mathbf{w}^l$) based on the clustering acquired in the previous step.
7. Train $K$ forecasting models for every aggregated time series and perform forecast one short window ahead. Sum up $K$ forecasts to create the final aggregated forecast.
8. Test $K$ aggregated time series for EDF change by the k-Sample Anderson-Darling test.
   - If change is detected (see Sect. 3.5), then remove the first (oldest) short data stream window and assign the newest acquired short data stream window to long window and go to step 2. Do step 4 with the range of the number of

clusters $\langle K-2, K+2\rangle$, where $K$ is the number of clusters in the previous iteration.
- If change is not detected then remove the first (oldest) short data stream window and assign the newest acquired short data stream window to long window and go to step 6.

## 4 Evaluation of ClipStream and FeaClip

The research questions that will be answered in the evaluation section are as follows:

1. Can the proposed multiple data stream clustering method *ClipStream* improve forecasting accuracy of aggregate electricity consumption against simple aggregation approach and other clustering benchmarks?
2. How sensitive is *ClipStream* on settings of its parameters?
3. How accurate is the proposed time series representation method *FeaClip* for clustering time series datasets against benchmark representations?
4. What is the computational complexity of the whole *ClipStream*?
5. How useful is *FeaClip* for high-performance computing purposes (i.e. measure time complexity against other benchmark methods)?

### 4.1 Improvement of forecasting accuracy by ClipStream

#### 4.1.1 Smart meter data

To evaluate the behaviour of *ClipStream* the clustering method alongside forecast accuracy performance, we used four different datasets gathered from smart meters that consist of a large number of variable patterns. Measurement data include London, Irish, and two types of Slovak electricity consumption data.

Irish data was collected by the Irish Commission for Energy Regulation (CER) and is available from Irish Social Science Data Archive.[1] These data contain three different types of customers: residential, SMEs and others. The largest group is residential, where after removing consumers with missing data, we have 3639 residential consumers left. The frequency of data measurements was every thirty minutes - during a day 48 measurements were performed.

London data was collected as a part of the Low Carbon London project involved over five thousand households in the London area (Schofield et al. 2015). Smart meter data, time-of-use tariff data, and survey data were collected to investigate the impacts of a wide range of low-carbon technologies on London's electricity distribution network. After removing consumers missing more than one day of data, we have 5066 residential consumers left. The frequency of data measurements was every thirty minutes - during a day 48 measurements were performed.

The first type of Slovak data was collected within the project "International Centre of Excellence for Research of Intelligent and Secure Information-Communi-cation Technologies and Systems". These measurements are obtained from Slovak enter-

[1] http://www.ucd.ie/issda/data/commissionforenergyregulationcer/.

prises and factories. After removing consumers with missing data, those with zero consumption and maximal consumption higher than 42 kW, the customer base comprised 3630 consumers.

The second type of Slovak data was available thanks to a public hackathon competition named EnergyHack[2] relating to smart meter data exploration, end-customer support, and renewable energy integration. Data collection consists, after removing consumers with missing data, of 989 SMEs consumers and were measured during the whole year of 2016. The frequency of both Slovak datasets was every fifteen minutes, so daily 96 measurements were performed. The frequency of data measurements was aggregated to half-hourly in order for it to be equal and comparable with the Irish and London datasets.

Differences between data from residences, SMEs and factories are evident. The amount of consumption of residences and SMEs consumers are similarly small and irregular, but they differ in weekly profiles; while on weekends SMEs are mostly closed, at residences consumers are then at home. Factories are slightly similar to SME data in relation to weekly profiles, but the amount of consumption can be very high and is mostly regular during the week and irregular during the year (i.e., a day off for the whole factory, a period of the year when central heating is turned on or air conditioning usage etc.). Therefore the interesting results of clustering alongside forecasting for these three categories were expected and prove the depth and sufficiency of the experiments.

#### 4.1.2 Benchmark methods

Our proposed *ClipStream* clustering approach was compared with two methods of aggregation of electricity consumption alongside forecasting. First is the simple summation method of all consumers into one time series. Second is batch clustering by three various methods.

The first is K-medoids with time series pre-processed by Discrete Fourier Transformation (DFT) and min-max normalisation. For faster computations, the Fast Fourier Transform (Cooley and Tukey 1965) (FFT) was used. For one batch (window), $u * 8$ coefficients were extracted from the Fourier transformation and then back inverted. For every new window, new Fourier representations of time series and an optimal number of clusters by the DB-index were computed. It follows similar methodology as in the paper of Beringer and Hüllermeier (2007).

The second is K-medoids with time series preprocessed by Multiple Linear Regression (MLR) and min-max normalisation. In our scenario, MLR extracts daily and weekly seasonal regression coefficients, so together for every long window $48 + 6$ regression coefficients were extracted. For every new window, new MLR representations of time series and an optimal number of clusters by the DB-index were computed. It follows similar methodology as in the papers of Laurinec and Lucká (2017, 2018b).

[2] http://energyhack.sk/?lang=en.

The third is k-Shape clustering algorithm with time series preprocessed by the PAA representation method and z-score normalisation. Z-score normalisation is defined as

$$x' = \frac{x - \mu}{\sigma}, \tag{8}$$

where $x$ is the original observation, $\mu$ is the arithmetic mean of the window, $\sigma$ is the standard deviation of the window, and $x'$ is the normalised observation. PAA extracts mean from predefined lengths of non-overlapping time series segments. In our scenario, this segment has length 6, so for every day 8 means are extracted, so it is equal to lengths of *FeaClip* and DFT representations. For every new window, new PAA representations of time series and an optimal number of clusters by the DB-index were computed. It follows similar methodology as in the paper of Jarábek et al. (2017).

#### 4.1.3 Reproducibility of results

The source code of the all implemented methods is available online.[3] Time series representations methods alongside *FeaClip*, DFT, MLR and PAA are available in the TSrepr package[4] that enables its fast computing (Laurinec 2018). The subset of London residential dataset, ready for clustering and forecasting, is available on OpenML datasets repository.[5]

#### 4.1.4 Results

We conducted many experiments on four different datasets to evaluate whether creating clustered electricity consumption time series by *ClipStream* outperforms benchmark methods (DFT representation clustered by K-medoids, k-Shape method, MLR representation clustered by K-medoids, and a simple aggregation of all consumers as described in Sect. 4.1.2) in the sense of improvement of forecast accuracy.

Three different periods of the year from Slovak factories and Irish residential datasets were chosen to investigate the performance of the implemented approaches. For the London residential dataset, five consecutive months were chosen because only a length of one year of data was at our disposal. For the Slovak SMEs dataset, four consecutive months were chosen because only the length of one year of data was at our disposal. The Slovak factories testing dataset contains three months of data measurements from 2013 and 2014 (23.9.2013–26.10.2013, 10.2.2014–11.3.2014 and 2.6.2014–1.7.2014), together 94 days. The Irish residential testing dataset contains three months of data measurements from 2010 (1.2.2010–28.2.2010, 1.5.2010–31.5.2010 and 1.8.2010–31.8.2010), together 90 days. The London residential testing dataset contains measurements from five months (132 days) of 2013 (21.5.2013–30.9.2013). Finally, the Slovak SMEs testing dataset contains measurements from four months (102 days) of 2016 (21.5.2016–31.8.2016). Moreover, we had additional

[3] https://github.com/PetoLau/ClipStream.

[4] https://cran.r-project.org/package=TSrepr.

[5] https://www.openml.org/d/41060.

**Table 2** Average daily MAPE (%) ± standard deviation of 6 forecasting methods evaluated on Slovak factories dataset

| MAPE | Slovak Factories | | | | |
|---|---|---|---|---|---|
| Method N. | Agg. | DFT | MLR | k-Shape | ClipStream |
| Cart | 3.151 ± 1.31 | 3.03 ± 1.26 | 3.045 ± 1.3 | **3.026 ± 1.27** | 3.037 ± 1.27 |
| Ctree.lag | 3.041 ± 1.25 | 2.916 ± 1.26 | 2.911 ± 1.3 | 2.916 ± 1.28 | **2.898 ± 1.27** |
| Ctree.dft | 3.104 ± 1.23 | 2.991 ± 1.24 | 3.002 ± 1.27 | 3.003 ± 1.25 | **2.979 ± 1.26** |
| Rf.lag | 2.912 ± 1.3 | 2.857 ± 1.27 | 2.868 ± 1.32 | 2.856 ± 1.3 | **2.851 ± 1.29** |
| Rf.dft | 2.986 ± 1.27 | 2.931 ± 1.26 | 2.938 ± 1.3 | 2.926 ± 1.28 | **2.922 ± 1.27** |
| Exp | 2.399 ± 1.19 | 2.406 ± 1.18 | **2.399 ± 1.18** | 2.400 ± 1.18 | 2.4 ± 1.18 |
| Mean | 2.932 ± 1.26 | 2.855 ± 1.25 | 2.861 ± 1.28 | 2.855 ± 1.26 | **2.848 ± 1.26** |

Values in bold represent best results among the all methods

21 days in front of every first day of the mentioned periods. Those days are used for clustering and training forecasting methods.

Forecasting accuracy performance of *ClipStream* and four benchmark methods and six forecasting methods is shown in Table 2 for the Slovak factories dataset, in Table 3 for the Irish residential dataset, in Table 4 for the Slovak SMEs dataset, and in Table 5 for the London residential dataset. The setting of parameters of *ClipStream* method for each dataset is shown in Table 6. In 23 cases from 24 *ClipStream* were better in terms of average daily MAPE than a simple aggregation method. In 18 cases from 24 *ClipStream* were better in terms of average daily MAPE than benchmark method using DFT coefficients. In 18 cases from 24 *ClipStream* were better in terms of average daily MAPE than benchmark method using MLR representation. In 22 cases from 24 *ClipStream* were better in terms of average daily MAPE than benchmark method using k-Shape clustering. In 16 cases from 24 *ClipStream* were better in terms of average daily MAPE than any of the benchmark methods.

The significance of forecasting results was tested by the Wilcoxon rank sum test, whereby statistical (alternative) hypothesis is formulated so that errors of forecasting alongside clustering of consumers with *ClipStream* are lower than forecasting errors achieved by the benchmark aggregation method. Results are shown in Table 7 for the Slovak factories dataset, in Table 8 for the Irish residential dataset, in Table 9 for the Slovak SMEs dataset, and in Table 10 for the London residential dataset. A significant improvement was marked in those cases where the $p$ value was lower than 0.05. In 19 cases from 24 *ClipStream* had significantly better forecasting accuracy than the simple aggregation method. In 10 cases from 24 *ClipStream* had significantly better forecasting accuracy than the benchmark method using DFT coefficients. In 6 cases from 24 *ClipStream* had significantly better forecasting accuracy than the benchmark method using MLR representation. In 10 cases from 24 *ClipStream* had significantly better forecasting accuracy than the benchmark method using k-Shape clustering.

The impact of outlier detection and change detection in clustering of *FeaClip* representation with K-medoids on forecasting accuracy results is shown in Tables 11 and 12. This evaluation incorporates experiments on the Slovak SMEs (Table 11) and London residential datasets (Table 12). Outlier detection procedure in clustering of *FeaClip*

**Table 3** Average daily MAPE (%) ± standard deviation of 6 forecasting methods evaluated on Irish residential dataset

| MAPE | IrishResidential | | | | |
|---|---|---|---|---|---|
| MethodN. | Agg. | DFT | MLR | k-Shape | ClipStream |
| Cart | 4.182 ± 2.36 | 4.072 ± 2.35 | 4.036 ± 2.33 | 4.041 ± 2.31 | **4.001 ± 2.28** |
| Ctree.lag | 4.102 ± 2.18 | 3.901 ± 2.23 | 3.862 ± 2.26 | 3.882 ± 2.27 | **3.843 ± 2.21** |
| Ctree.dft | 4.069 ± 2.17 | 3.895 ± 2.23 | 3.86 ± 2.2 | 3.926 ± 2.17 | **3.823 ± 2.19** |
| Rf.lag | 3.839 ± 2.21 | 3.796 ± 2.22 | 3.771 ± 2.21 | 3.797 ± 2.19 | **3.762 ± 2.21** |
| Rf.dft | 3.801 ± 2.21 | 3.773 ± 2.21 | 3.74 ± 2.2 | 3.769 ± 2.18 | **3.724 ± 2.19** |
| Exp | 4.786 ± 2.43 | 4.734 ± 2.36 | 4.707 ± 2.38 | 4.75 ± 2.44 | **4.631 ± 2.28** |
| Mean | 4.13 ± 2.26 | 4.028 ± 2.27 | 3.996 ± 2.26 | 4.027 ± 2.26 | **3.964 ± 2.23** |

Values in bold represent best results among the all methods

**Table 4** Average daily MAPE (%) ± standard deviation of 6 forecasting methods evaluated on Slovak SMEs dataset

| MAPE | SlovakSME | | | | |
|---|---|---|---|---|---|
| MethodN. | Agg. | DFT | MLR | k-Shape | ClipStream |
| Cart | 4.787 ± 3.58 | 4.546 ± 3.52 | 4.531 ± 3.49 | 4.566 ± 3.54 | **4.496 ± 3.48** |
| Ctree.lag | 4.876 ± 3.56 | 4.426 ± 3.55 | 4.452 ± 3.51 | 4.491 ± 3.56 | **4.375 ± 3.5** |
| Ctree.dft | 4.937 ± 3.59 | 4.518 ± 3.58 | 4.504 ± 3.57 | 4.616 ± 3.31 | **4.472 ± 3.57** |
| Rf.lag | 4.56 ± 3.62 | 4.331 ± 3.53 | 4.329 ± 3.5 | 4.36 ± 3.56 | **4.291 ± 3.5** |
| Rf.dft | 4.666 ± 3.61 | 4.408 ± 3.56 | 4.4 ± 3.53 | 4.465 ± 3.59 | **4.36 ± 3.52** |
| Exp | 4.265 ± 2.99 | 3.674 ± 2.7 | **3.657 ± 2.7** | 3.667 ± 2.7 | 3.677 ± 2.71 |
| Mean | 4.682 ± 3.49 | 4.317 ± 3.41 | 4.312 ± 3.38 | 4.361 ± 3.43 | **4.278 ± 3.38** |

Values in bold represent best results among the all methods

**Table 5** Average daily MAPE (%) ± standard deviation of 6 forecasting methods evaluated on London residential dataset

| MAPE | LondonResidential | | | | |
|---|---|---|---|---|---|
| MethodN. | Agg. | DFT | MLR | k-Shape | ClipStream |
| Cart | 4.179 ± 1.71 | **3.956 ± 1.73** | 3.979 ± 1.75 | 4.012 ± 1.75 | 3.969 ± 1.75 |
| Ctree.lag | 4.094 ± 1.85 | 3.886 ± 1.87 | **3.85 ± 1.88** | 3.876 ± 1.85 | 3.856 ± 1.87 |
| Ctree.dft | 4.139 ± 1.74 | 3.911 ± 1.77 | **3.899 ± 1.77** | 3.955 ± 1.78 | 3.919 ± 1.77 |
| Rf.lag | 3.808 ± 1.8 | 3.778 ± 1.82 | **3.77 ± 1.84** | 3.797 ± 1.82 | 3.783 ± 1.82 |
| Rf.dft | 3.857 ± 1.75 | 3.823 ± 1.76 | **3.821 ± 1.77** | 3.847 ± 1.76 | 3.834 ± 1.76 |
| Exp | 4.269 ± 2.21 | 4.223 ± 2.17 | 4.088 ± 2.07 | 4.109 ± 2.09 | **4.086 ± 2.08** |
| Mean | 4.058 ± 1.84 | 3.93 ± 1.85 | **3.901 ± 1.85** | 3.933 ± 1.84 | 3.908 ± 1.84 |

Values in bold represent best results among the all methods

**Table 6** Statistics of the behaviour of *ClipStream* on four different datasets from smart meters. n. refers to number, K to a number of clusters and ch.d. to change detections

| | Slovak Factories | Slovak SME | Irish Resid. | London Resid. |
|---|---|---|---|---|
| N. of consumers | 3630 | 989 | 3639 | 5066 |
| Long/short win. length | 1008/48 | 1008/48 | 1008/48 | 1008/48 |
| $\lambda$ | 1 | 1.25 | 1.5 | 1.5 |
| $\alpha$ | 0.0005 | 0.05 | 0.05 | 0.05 |
| $K_{min}$ | 8 | 28 | 8 | 20 |
| $K_{max}$ | 16 | 38 | 16 | 30 |
| Average n. of K | 8.91 | 31.85 | 12.77 | 25.62 |
| Average n. of outliers | 632.06 | 76.80 | 256.10 | 351.21 |
| Average n. of ch.d. | 2.78 | 25.33 | 4.09 | 13.94 |
| Ratio of ch.d. | 0.35 | 1.00 | 0.42 | 0.68 |

$K_{min}$ and $K_{max}$ are a minimal and maximal number of possible clusters (i.e., borders)

**Table 7** $p$ values of comparison between the *ClipStream* and all the benchmarks on Slovak factories dataset

| $p$ value | Slovak Factories | | | |
|---|---|---|---|---|
| Method N. | ClipS.-Agg. | ClipS.-DFT | ClipS.-MLR | ClipS.-k-Shape |
| Cart | **0.0003** | 0.6999 | 0.4669 | 0.5045 |
| Ctree.lag | $<$ **0.0001** | 0.1464 | 0.1404 | 0.2120 |
| Ctree.dft | **0.0001** | 0.1653 | **0.0297** | **0.0331** |
| Rf.lag | **0.0024** | 0.4609 | **0.0407** | 0.2769 |
| Rf.dft | **0.0003** | 0.4076 | **0.0438** | 0.1878 |
| Exp | 0.6691 | 0.4076 | 0.6300 | 0.7129 |

Bold values represent significant improvements of the *ClipStream* with the combination of the given forecasting method

**Table 8** $p$ values of comparison between the *ClipStream* and all the benchmarks on Irish residential dataset

| $p$ value | Irish Residential | | | |
|---|---|---|---|---|
| Method N. | ClipS.-Agg. | ClipS.-DFT | ClipS.-MLR | ClipS.-k-Shape |
| Cart | **0.0002** | **0.0019** | 0.1592 | 0.0832 |
| Ctree.lag | $<$ **0.0001** | **0.0178** | 0.2271 | 0.0982 |
| Ctree.dft | $<$ **0.0001** | **0.0052** | 0.0783 | **0.0003** |
| Rf.lag | 0.1742 | 0.0715 | 0.3452 | **0.0234** |
| Rf.dft | 0.2036 | **0.0081** | 0.1413 | **0.0039** |
| Exp | **0.0178** | **0.0444** | 0.3247 | 0.0710 |

Bold values represent significant improvements of the *ClipStream* with the combination of the given forecasting method

**Table 9** $p$ values of comparison between the *ClipStream* and all the benchmarks on Slovak SMEs dataset

| $p$ value | Slovak SME | | | |
|---|---|---|---|---|
| Method N. | ClipS.-Agg. | ClipS.-DFT | ClipS.-MLR | ClipS.-k-Shape |
| Cart | $<$**0.0001** | **0.0398** | 0.1005 | **0.0071** |
| Ctree.lag | $<$**0.0001** | 0.1699 | **0.0087** | **0.0001** |
| Ctree.dft | $<$**0.0001** | **0.0342** | 0.0914 | $<$**0.0001** |
| Rf.lag | $<$**0.0001** | 0.1115 | **0.0308** | **0.0054** |
| Rf.dft | $<$**0.0001** | **0.030** | **0.043** | **0.0008** |
| Exp | $<$**0.0001** | 0.6168 | 0.9258 | 0.882 |

Bold values represent significant improvements of the *ClipStream* with the combination of the given forecasting method

**Table 10** $p$ values of comparison between the *ClipStream* and all the benchmarks on London residential dataset

| $p$ value | London Residential | | | |
|---|---|---|---|---|
| Method N. | ClipS.-Agg. | ClipS.-DFT | ClipS.-MLR | ClipS.-k-Shape |
| Cart | $<$**0.0001** | 0.7993 | 0.3171 | **0.0168** |
| Ctree.lag | $<$**0.0001** | **0.0283** | 0.7335 | 0.1845 |
| Ctree.dft | $<$**0.0001** | 0.6997 | 0.8948 | 0.0528 |
| Rf.lag | 0.3179 | 0.7365 | 0.9467 | 0.0722 |
| Rf.dft | 0.4950 | 0.8118 | 0.8634 | 0.1052 |
| Exp | $<$**0.0001** | **0.0002** | 0.7305 | 0.1199 |

Bold values represent significant improvements of the *ClipStream* with the combination of the given forecasting method

representation improved forecasting accuracy in terms of average daily MAPE in 11 cases from 12 in comparison with simple clustering of *FeaClip* representation. Change detection procedure in clustering of *FeaClip* representation with outlier detection procedure improved forecasting accuracy in terms of average daily MAPE in 11 cases from 12 in comparison with clustering of *FeaClip* representation with outlier detection procedure. These experiments proved the usefulness of both procedures not just for acquiring interpretable insights during data streams clustering, but also for more accurate forecasting.

The evaluation of the sensitivity to parameter settings in the proposed *ClipStream* method in forecasting results is shown in Fig. 10. This evaluation was again performed on the Slovak SMEs and London residential datasets. The sensitivity to $\alpha$ in change detection, $\lambda$ in outlier detection, the range of the number of clusters and the length of long data stream window was tested. The change of the length of data stream window had the biggest impact on forecasting accuracy, where the increase of the length from 21 days to 28 days worsened results significantly. The other three parameters do not have such a great impact on forecasting accuracy results.

The investigation of multiple data streams clustering processes through time is important for evaluating the stability of the used clustering algorithm and also for

**Table 11** Average daily MAPE (%) ± standard deviation of 6 forecasting methods with combination of simple *FeaClip* clustering, outlier detection, and *ClipStream* evaluated on Slovak SMEs dataset

| MAPE | Slovak SME | | |
|---|---|---|---|
| Method N. | FeaClip | FeaClip+Out.Det. | ClipStream |
| Cart | 4.603 ± 3.57 | 4.591 ± 3.53 | **4.572 ± 3.49** |
| Ctree.lag | 4.542 ± 3.64 | 4.48 ± 3.57 | **4.478 ± 3.53** |
| Ctree.dft | 4.654 ± 3.65 | 4.581 ± 3.6 | **4.576 ± 3.54** |
| Rf.lag | 4.414 ± 3.62 | 4.38 ± 3.56 | **4.377 ± 3.52** |
| Rf.dft | 4.494 ± 3.62 | **4.447 ± 3.58** | 4.453 ± 3.52 |
| Exp | 3.789 ± 2.63 | **3.668 ± 2.7** | **3.668 ± 2.72** |
| Mean | 4.416 ± 3.46 | 4.358 ± 3.42 | **4.354 ± 3.39** |

Values in bold represent best results among the all methods

**Table 12** Average daily MAPE (%) ± standard deviation of 6 forecasting methods with combination of simple *FeaClip* clustering, outlier detection, and *ClipStream* evaluated on London residential dataset

| MAPE | London Residential | | |
|---|---|---|---|
| Method N. | FeaClip | FeaClip+Out.Det. | ClipStream |
| Cart | 4.002 ± 1.73 | 3.982 ± 1.72 | **3.969 ± 1.75** |
| Ctree.lag | 3.872 ± 1.87 | 3.864 ± 1.84 | **3.856 ± 1.87** |
| Ctree.dft | 3.925 ± 1.76 | 3.929 ± 1.75 | **3.919 ± 1.77** |
| Rf.lag | 3.794 ± 1.82 | 3.785 ± 1.81 | **3.783 ± 1.82** |
| Rf.dft | 3.844 ± 1.76 | 3.835 ± 1.75 | **3.834 ± 1.76** |
| Exp | 4.108 ± 2.1 | 4.097 ± 2.07 | **4.086 ± 2.08** |
| Mean | 3.924 ± 1.84 | 3.915 ± 1.82 | **3.908 ± 1.84** |

Values in bold represent best results among the all methods

energy practitioners for analysing and monitoring the smart grid (consumers). *ClipStream* behaviour statistics during training are shown in Table 6 and data streams clustering process in a time series form is visualised in Fig. 11. Dynamic changes of the number of clusters, the number of outliers and the number of change detections are meant by the behaviour of our proposed clustering data streams approach. Set borders (minimal-maximal) of a number of clusters for all clustering methods (*ClipStream* and benchmarks) and four datasets are also shown on Table 6.

From the statistics, we can see interesting differences among the four datasets. At first, it has to be noted that *ClipStream* parameters were set optimally for improving forecasting accuracy. For the Slovak factories dataset, it was optimal to set $\lambda$ small (equal to 1), so a large amount of outliers were always detected. This setting improved forecasting performance and clustering as well. For the Slovak SMEs dataset, it was optimal to set a high number of clusters. This also implied a large number of change detections. This is due to the fact that a small number of consumers are in disposal for the Slovak SMEs dataset, therefore the aggregated time series fluctuated and were noisy. For the Slovak factories and Irish residential datasets, the small ratio of re-

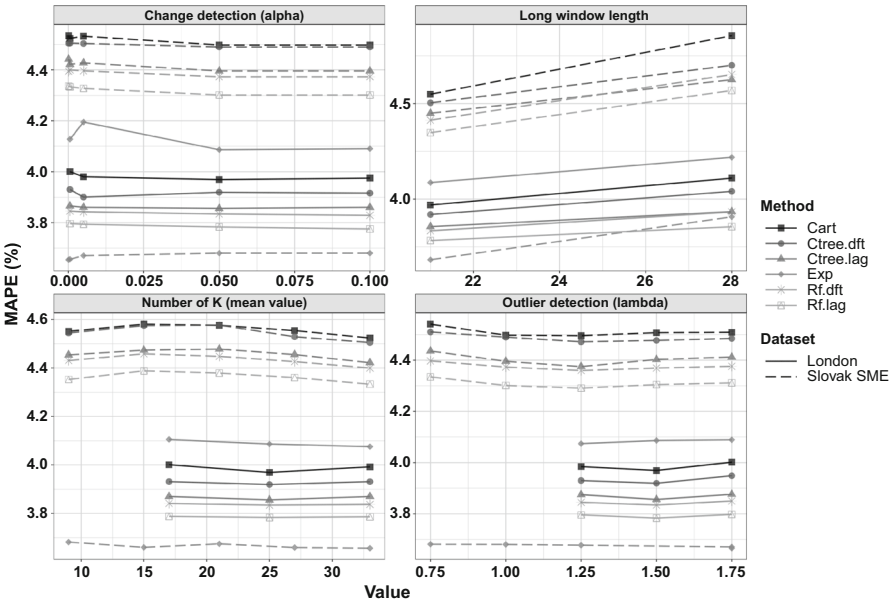


**Fig. 10** The evaluation of *ClipStream* on sensitivity to its parameters

clustering was needed for optimal forecasting performance. It suggests that created clusters were satisfactory for a longer time.

On the Fig. 11 visualisation, these differences and amount of change detections can be seen.

### 4.2 Evaluation of FeaClip representation on other datasets

To investigate different applications of *FeaClip* representation with respect to a type of dataset where this representation can be used, experiments on a large repository of time series datasets were conducted. This repository is called UCR time series classification archive (Chen et al. 2015) and consists of 85 datasets appropriate for testing classification methods with various distance measures and representations of time series. Clustering methods can be also evaluated on these datasets because true labels are available, so the external validation measure of clustering can be applied.

The Fowlkes–Mallows index external clustering validation measure was chosen to evaluate results because it has more meaningful values in the case of unrelated clusterings with a large amount of data or in the case of added noise in the dataset than the Rand index (Fowlkes and Mallows 1983). The Fowlkes–Mallows index is defined as:

$$FM = \sqrt{\frac{TP}{TP + FP} \cdot \frac{TP}{TP + FN}},$$

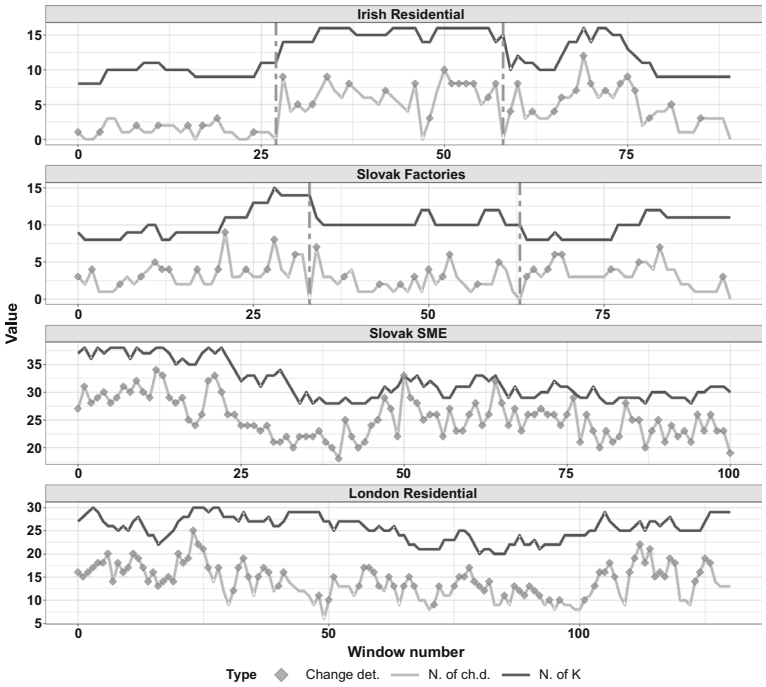


**Fig. 11** The behaviour of *ClipStream* on four different datasets. N. of ch.d. refers to the number of change detections, and N. of K to a number of clusters. Points in plots labeled as Change det. represents change detection and the following update of clustering. Vertical gray dashed lines border different periods during the year

where $TP$ is the number of true positives, $FP$ is the number of false positives, and $FN$ is the number of false negatives between the real classification and the calculated clustering.

The proposed *FeaClip* representation of time series clustered by K-medoids was compared with the clustered representations by DFT and the original time series. The first eight DFT coefficients were always picked to have the same length as the *FeaClip* representation. Datasets in the UCR archive are divided to train and test parts, so we used the training dataset to perform K-medoid clustering and obtain medoids. Time series (or representations) from the test dataset were then classified to the nearest medoid so labels from the clustering were obtained for comparison with true labels from the test set.

Clustering with *FeaClip* representation performed better than both benchmarks in 31 from 85 occasions and on 20 more datasets it was better at least than one benchmark method. Values of Fowlkes-Mallows index on 31 datasets where *FeaClip* representation outperformed benchmarks are shown in Fig. 12.

The proposed *FeaClip* representation helps create more accurate clusterings on many other types of datasets as well, however, it is clear that *FeaClip* is not the best representation for every type of time series in general. Experiments suggest that the

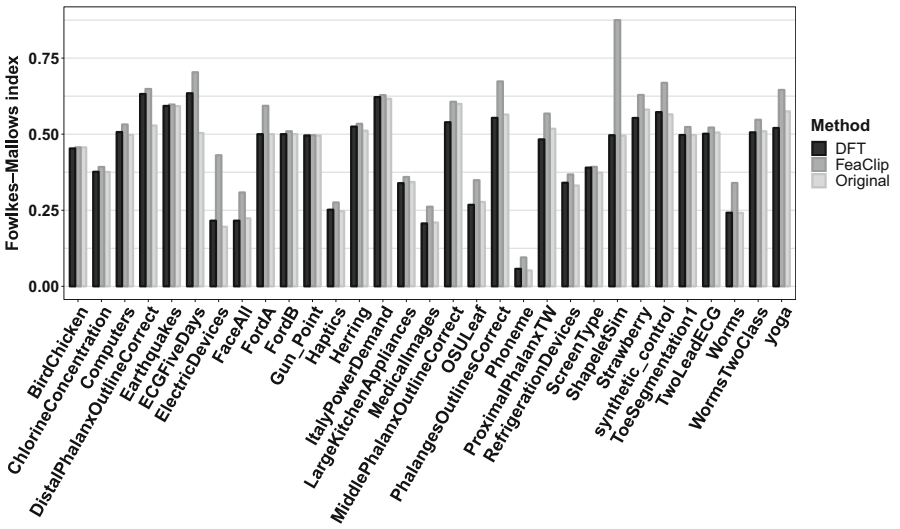


**Fig. 12** The clustering performance of three representations on UCR archive datasets where *FeaClip* outperformed benchmark methods. The names of datasets are identical as in the UCR repository

application of *FeaClip* is also proper in other types of sensor data such as ECG or appliances usage.

### 4.3 Computational complexity

The *FeaClip* representation computational complexity is linear $O(n)$, where $n$ is the length of the data window. This is due to the computation of average from a data window at the beginning, other computations as RLE and features extraction have lower complexity. Our benchmark methods used in experiments are FFT with computational complexity $O(n \log n)$, PAA with $O(n)$, and MLR with $O(d^2 n)$, where $d$ is the number of parameters (features). Execution time comparison of *FeaClip*, FFT and PAA on randomly generated data from a normal distribution of lengths $2^i$, $i = 3, \ldots, 14$, is shown in Fig. 13. As expected *FeaClip* with an increasing amount of data having lower execution time as FFT, proves the usability of *FeaClip* representation of data streams for high-performance computing. Execution times of PAA and *FeaClip* are similar (since both have $O(n)$) with slightly better results of the proposed *FeaClip*. MLR was omitted from testing because it had significantly worse results than all other methods.

Now, let us decompose the whole *ClipStream* procedure and discuss corresponding computational complexities. The data abstraction step consists of the representation computation that has mentioned $O(n)$ ($N$-times computed) and outlier detection that has $O(N)$ complexity, where $N$ is the number of data streams. The offline phase consists of the PAM clustering algorithm that for each iteration has the quadratic complexity of $O(K((N - o) - K)^2)$, where $K$ is a number of clusters and $o$ is a number of outliers. It is executed maximally five times $\langle K - 2, K + 2 \rangle$ as it is

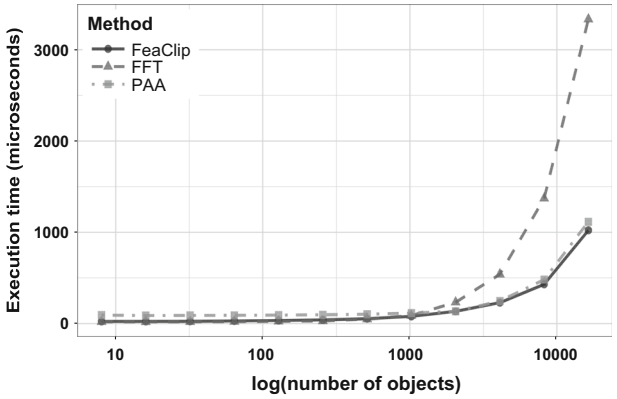


**Fig. 13** Execution time of *FeaClip*, FFT and PAA representation methods

mentioned in Sect. 3.3. Outliers assignment to nearest medoids takes $O(oK)$. The computation of k-sample Anderson-Darling test statistic has complexity $O(n)$. However, our change detection method can lower the computational load of *ClipStream* dramatically. On the one hand, our benchmark clustering method k-Shape has time complexity $O(\max\{NKnlog(n), Nn^2, Kn^3\})$ (Paparrizos and Gravano 2015). If the time series (data stream windows) are long, then the time complexity of k-Shape can be very high. It is obvious that the bottleneck of our approach is in clustering filtered representations that have quadratic complexity. However, the PAM algorithm can be accelerated by parallel computation of the dissimilarity matrix before it inputs the algorithm and also PAM itself can be parallelised to speed up computations. Computations of *FeaClip* representations, outlier detection, outlier assignment, and also the change detection step can be simply parallelised to speed up computations where the number of consumers is very large or in the case of long data streams windows.

## 5 Discussion

The evaluation showed that methods used in our previous works (model-based representation MLR) also had very good forecasting results here. However, the proposed approach using *FeaClip* representation in combination with automatic outlier and change detection, even improved forecasting performance in most cases. In addition, the analysis in Sect. 4.3 proved the usefulness of the proposed approach for high-performance computing, in opposition to all the benchmarks.

We proposed the new outlier detection method for seasonal time series streams that can be further studied and evaluated in the data mining community. Experiments in Sect. 4.2 also proved the usefulness of the proposed time series representation *FeaClip* on other datasets. Using statistical k-sample tests for EDF change detection also has further potential in data stream mining scenarios.

For practitioners who forecast electricity consumption, or other multiple seasonal time series, we showed the stability of improvement of forecasting accuracy using

*ClipStream*. *ClipStream* also offers detections of abnormal behaviours, changes in behaviour, and extraction of typical patterns in data through clustering. *ClipStream* results also show that it is not very sensitive to parameter setting (defaults are performing well).

## 6 Conclusion

In our paper, we have proposed the interpretable *ClipStream* approach for multiple data stream clustering of smart meter data. Results of clustering were used for analysing consumers of electricity consumption and for improving forecasting accuracy. The online phase of the *ClipStream* process consists of a newly proposed extraction of interpretable features from the clipped representation named *FeaClip* and a fast automatic outlier detection directly from the most important features of *FeaClip*. The offline phase consists of clustering filtered representations by K-medoids, aggregating time series within clusters to create more predictable groups of consumers and the EDF change detection procedure used for decreasing computational load. *ClipStream* clustering alongside forecasting was compared with four benchmark methods: simple aggregation of consumption, clustering approach using DFT coefficients or regression coefficients, and the k-Shape clustering method. Six forecasting methods with five aggregation combination methods were evaluated on four different smart meter datasets. In 16 cases from 24 the *ClipStream* approach was better in the mean of forecasting accuracy than all benchmark methods. The significance of forecasting results was also tested and proved the stability of the proposed approach. The analysis of the behaviour of *ClipStream* on variable datasets proved the adaptability of the proposed approach with respect to detecting outliers or change detection in data streams.

We evaluated the suitability of *FeaClip* representation itself for clustering time series on 85 publicly available datasets. On 31 occasions *FeaClip*, was better than benchmark methods, which suggests other interesting applications of the proposed feature extraction technique from the clipped representation. We discussed and experimentally compared the computational complexity of *FeaClip* with the FFT and PAA methods, in which *FeaClip* demonstrated suitability for high-performance computing.

In future work, it would be interesting to investigate modifications of the proposed *FeaClip* representation. For example, incorporating feature extraction from multiple lengths of windows from the data stream for capturing multiple seasonalities, trend changing and concept drifts (i.e., complex behaviour of a data stream). Usage of a faster clustering algorithm than PAM like a fast density-based algorithm would be beneficial; unfortunately, current state-of-the-art density-based clustering methods for time series are limited.

**Acknowledgements** This work was partially supported by the Slovak Research and Development Agency, Grant Nos. APVV-16-0484 and APVV-16-0213, and the Scientific Grant Agency of The Slovak Republic, Grant No. VG 1/0458/18.

## References


Aggarwal CC, Han J, Wang J, Yu PS (2003) A framework for clustering evolving data streams. In: Proceedings of the 29th international conference on Very large data bases-volume 29, VLDB Endowment, pp 81–92

Aghabozorgi S, Seyed Shirkhorshidi A, Ying Wah T (2015) Time-series clustering: a decade review. Inf Syst 53:16–38

Amini A, Saboohi H, Herawan T, Wah TY (2016) Mudi-stream: a multi density clustering algorithm for evolving data stream. J Netw Comput Appl 59:370–385

Appice A, Guccione P, Malerba D, Ciampi A (2014) Dealing with temporal and spatial correlations to classify outliers in geophysical data streams. Inf Sci 285:162–180

Arora P, Deepali Varshney S (2016) Analysis of k-means and k-medoids algorithm for big data. Procedia Comput Sci 78:507–512

Bagnall A, Ratanamahatana C, Keogh E, Lonardi S, Janacek G (2006) A bit level representation for time series data mining with shape based similarity. Data Min Knowl Discov 13(1):11–40

Beringer J, Hüllermeier E (2007) Fuzzy clustering of parallel data streams. In: Advances in fuzzy clustering and its application, pp 333–352

Breiman L (2001) Random forests. Mach Learn 45(1):5–32

Breiman L, Friedman J, Stone CJ, Olshen RA (1984) Classification and regression trees. CRC Press, Amsterdam

Chan KP, Fu AWC (1999) Efficient time series matching by wavelets. In: Data engineering, 1999. Proceedings., 15th international conference on, IEEE, pp 126–133

Chen JY, He HH (2016) A fast density-based data stream clustering algorithm with cluster centers self-determined for mixed data. Inf Sci 345:271–293

Chen L, Zou LJ, Tu L (2012) A clustering algorithm for multiple data streams based on spectral component similarity. Inf Sci 183(1):35–47

Chen Y (2009) Clustering parallel data streams. InTech

Chen Y, Tu L (2007) Density-based clustering for real-time stream data. In: Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 133–142

Chen Y, Keogh E, Hu B, Begum N, Bagnall A, Mueen A, Batista G (2015) The ucr time series classification archive www.cs.ucr.edu/~eamonn/time_series_data

Cleveland RB, Cleveland WS, McRae JE, Terpenning I (1990) STL: a seasonal-trend decomposition procedure based on loess. J Off Stat 6(1):3–73

Cooley JW, Tukey JW (1965) An algorithm for the machine calculation of complex fourier series. Math Comput 19(90):297–301

Corduas M, Piccolo D (2008) Time series clustering and classification by the autoregressive metric. Comput Stat Data Anal 52(4):1860–1872

Dai BR, Huang JW, Yeh MY, Chen MS (2006) Adaptive clustering for multiple evolving streams. IEEE Trans Knowl Data Eng 18(9):1166–1180

Davies DL, Bouldin DW (1979) A cluster separation measure. IEEE Trans Pattern Anal Mach Intell 1(2):224–227

Esling P, Agon C (2012) Time-series data mining. ACM Comput Surv 45(1):1–34

Faloutsos C, Ranganathan M, Manolopoulos Y (1994) Fast subsequence matching in time-series databases. In: Proceedings of the 1994 ACM SIGMOD international conference on management of data, ACM, New York, SIGMOD '94, pp 419–429. https://doi.org/10.1145/191839.191925

Fowlkes EB, Mallows CL (1983) A method for comparing two hierarchical clusterings. J Am Stat Assoc 78(383):553–569

Gama J, Rodrigues PP (2007) Stream-based electricity load forecast. In: Proceedings of the 11th European conference on principles and practice of knowledge discovery in databases (PKDD 2007) vol 4702, pp 446–453

Hahsler M, Bolaños M (2016) Clustering data streams based on shared density between micro-clusters. IEEE Trans Knowl Data Eng 28(6):1449–1461

Hyndman R, Khandakar Y (2008) Automatic time series forecasting: the forecast package for R. J Stat Softw 27(3):1–22

Hyndman R, Koehler AB, Ord JK, Snyder RD (2008) Forecasting with exponential smoothing: the state space approach. Springer, Berlin

Jarábek T, Laurinec P, Lucká M (2017) Energy load forecast using s2s deep neural networks with k-shape clustering. In: Informatics, 2017 IEEE 14th international scientific conference on, IEEE, pp 140–145
Kaufman L, Rousseeuw P (2009) Finding groups in data: an introduction to cluster analysis. Wiley, London
Keogh E, Chakrabarti K, Pazzani M, Mehrotra S (2001) Locally adaptive dimensionality reduction for indexing large time series databases. In: Proceedings of the 2001 ACM SIGMOD international conference on management of data. ACM, New York, SIGMOD '01, pp 151–162. https://doi.org/10.1145/375663.375680
Keogh EJ, Pazzani MJ (1998) An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In: Proceedings of the fourth international conference on knowledge discovery and data mining. AAAI Press, KDD'98, pp 239–243
Keogh EJ, Pazzani MJ (2000) A simple dimensionality reduction technique for fast similarity search in large time series databases. In: Terano T, Liu H, Chen ALP (eds) Knowledge discovery and data mining. Current issues and new applications. Springer, Berlin, pp 122–133
Khan I, Huang JZ, Ivanov K (2016) Incremental density-based ensemble clustering over evolving data streams. Neurocomputing 191(Supplement C):34–43
Laurinec P (2018) TSrepr R package: time series representations. J Open Source Softw 3(23):577. https://doi.org/10.21105/joss.00577
Laurinec P, Lucká M (2016) Comparison of representations of time series for clustering smart meter data. In: Lecture notes in engineering and computer science: proceedings of the world congress on engineering and computer science 2016, pp 458–463
Laurinec P, Lucká M (2017) New clustering-based forecasting method for disaggregated end-consumer electricity load using smart grid data. In: 2017 IEEE 14th international scientific conference on informatics, pp 210–215, https://doi.org/10.1109/INFORMATICS.2017.8327248
Laurinec P, Lucká M (2018) Clustering-based forecasting method for individual consumers electricity load using time series representations. Open Comput Sci 8(1):38–50
Laurinec P, Lucká M (2018) Usefulness of unsupervised ensemble learning methods for time series forecasting of aggregated or clustered load. In: Appice A, Loglisci C, Manco G, Masciari E, Ras ZW (eds) New frontiers in mining complex patterns. Springer, Cham, pp 122–137
Laurinec P, Lóderer M, Vrablecová P, Lucká M, Rozinajová V, Ezzeddine AB (2016) Adaptive time series forecasting of energy consumption using optimized cluster analysis. In: Data mining workshops (ICDMW), 2016 IEEE 16th international conference on, IEEE, pp 398–405
Lin J, Keogh E, Lonardi S, Chiu B (2003) A symbolic representation of time series, with implications for streaming algorithms. In: Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery—DMKD '03 p 2. https://doi.org/10.1145/882085.882086
Livera AMD, Hyndman RJ, Snyder RD (2011) Forecasting time series with complex seasonal patterns using exponential smoothing. J Am Stat Assoc 106(496):1513–1527. https://doi.org/10.1198/jasa.2011.tm09771
Manjoro WS, Dhakar M, Chaurasia BK (2016) Operational analysis of k-medoids and k-means algorithms on noisy data. In: 2016 International conference on communication and signal processing (ICCSP), pp 1500–1505. https://doi.org/10.1109/ICCSP.2016.7754408
McGill R, Tukey JW, Larsen WA (1978) Variations of box plots. Am Stat 32(1):12–16
Paparrizos J, Gravano L (2015) k-shape: efficient and accurate clustering of time series. In: Proceedings of the 2015 ACM SIGMOD international conference on management of data, ACM, New York, SIGMOD '15, pp 1855–1870. https://doi.org/10.1145/2723372.2737793
Pereira CMM, de Mello RF (2014) TS-stream: clustering time series on data streams. J Intell Inf Syst 42(3):531–566
Pravilovic S, Bilancia M, Appice A, Malerba D (2017) Using multiple time series analysis for geosensor data forecasting. Inf Sci 380:31–52
Ratanamahatana C, Keogh E, Bagnall AJ, Lonardi S (2005) A novel bit level time series representation with implication of similarity search and clustering. In: Pacific-Asia conference on knowledge discovery and data mining. Springer, Berlin, pp 771–777
Razali NM, Wah YB et al (2011) Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. J Stat Model Anal 2(1):21–33
Rodrigues PP, Gama J, Pedroso J (2008) Hierarchical clustering of time-series data streams. IEEE Trans Knowl Data Eng 20(5):615–627
Schofield JR, Carmichael R, Tindemans S, Bilton M, Woolf M, Strbac G, et al (2015) Low carbon london project: data from the dynamic time-of-use electricity pricing trial, 2013

Scholz FW, Stephens MA (1987) K-sample anderson–darling tests. J Am Stat Assoc 82(399):918–924
Silva JA, Faria ER, Barros RC, Hruschka ER, Carvalho ACPLFD, Gama J (2013) Data stream clustering: a survey. ACM Comput Surv 46(1):1–31
Strasser H, Weber C (1999) On the asymptotic theory of permutation statistics. In: SFB adaptive information systems and modelling in economics and management science
Yang J, Ning C, Deb C, Zhang F, Cheong D, Lee SE, Sekhar C, Tham KW (2017) k-shape clustering algorithm for building energy usage patterns analysis and forecasting model accuracy improvement. Energy Build 146:27–37