Clustering Analysis of Commercial Vehicles Using Automatically Extracted Features from Time Series Data

Jordan Perr-Sauer, Adam Duran, and Caleb Phillips

National Renewable Energy Laboratory

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Technical Report
NREL/TP-2C00-74212
January 2020

Clustering Analysis of Commercial Vehicles Using Automatically Extracted Features from Time Series Data

Jordan Perr-Sauer, Adam Duran, and Caleb Phillips

National Renewable Energy Laboratory

Suggested Citation
Perr-Sauer, Jordan, Adam Duran, and Caleb Phillips. 2020. *Clustering Analysis of Commercial Vehicles Using Automatically Extracted Features from Time Series Data*. Golden, CO: National Renewable Energy Laboratory. NREL/TP-2C00-74212. https://www.nrel.gov/docs/fy20osti/74212.pdf.

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Technical Report
NREL/TP-2C00-74212
January 2020

National Renewable Energy Laboratory
15013 Denver West Parkway
Golden, CO 80401
303-275-3000 • www.nrel.gov

NOTICE

This work was authored by the National Renewable Energy Laboratory (NREL), operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. This work was supported by the Laboratory Directed Research and Development (LDRD) Program at NREL. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government.

Cover Photos by Dennis Schroeder: (clockwise, left to right) NREL 51934, NREL 45897, NREL 42160, NREL 45891, NREL 48097, NREL 46526.

Executive Summary

Standard of practice approaches to time series cluster analysis involve careful feature engineering, often utilizing expert input to tune and select features by hand. In many cases, expert input may not be readily available, or there may not yet exist a community consensus on the ideal features for a given application. This paper compares the results of several cluster analysis methods, using both hand selected features and those extracted automatically, when applied to large geospatial time series telematics data from commercial trucking fleets. The impacts of feature selection, dimensionality reduction, and choice of clustering algorithm on the quality of clustering results are explored. Results from this analysis confirm prior results that domain agnostic features are competitive with the hand engineered features with respect to clustering quality metrics. These results also provide new insight into the most successful strategies for identifying structure in large unstructured vehicle telematics data, and suggest that time series clustering using automatic feature extraction can be an effective approach to extract structure from large scale geospatial time series data in cases when hand selected features are not available.

Acknowledgments

This work was authored by Alliance for Sustainable Energy, LLC, the manager and operator of the National Renewable Energy Laboratory for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Vehicle Technologies Office. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

Acronym List

DOE	U.S. Department of Energy
EPA	U.S. Environmental Protection Agency
PCA	Principal Component Analysis
t-SNE	t-distributed Sochastic Neighbor Embedding
DTW	Dynamic Time Warping
JSON	Java Script Object Notation
RDD	Resilient Distributed Datasets
API	Application Programmer Interface
UDF	User Defined Function
DBSCAN	Density-based Spatial Clustering of Applications with Noise
KI	Kinetic Intensity
NREL	National Renewable Energy Laboratory

Table of Contents

List of Figures

List of Tables

1 Introduction

Industrial systems such as commercial vehicles are increasingly instrumented with sensors to monitor their health, status, and location. These data are becoming more prevalent. As volumes increase, automatic and computationally scalable methods for discovering the underlying structure are becoming increasingly valuable. Effectively using these data can provide critical operational insights such as the identification of anomalous events or the early symptoms of subsystem failure. This paper considers an approach to clustering of time series data using automatically extracted features in the domain of vehicle telematics data analysis and engages a distributed computing environment using "big data" technologies to ensure that the approaches considered can be used on datasets with arbitrarily many vehicles.

Applications of time series clustering are present in many industries. Anomaly detection of industrial sensor data can be used for early detection of mechanical failure and security threats. At the National Renewable Energy Laboratory (NREL) time series data is used in this way in multiple projects. For example, failure of the gearbox in a wind turbine can be predicted by analyzing time series data from the sensors on the turbine [15]. In another project sensors on NREL's supercomputer, Peregrine, produce time series data which can be used to optimize job scheduling such that energy consumption and heat exhaust are minimized [4].

This paper explores one aspect of time series clustering, feature extraction, using a real world geospatial time series dataset collected by telematics systems on commercial trucking fleets for NREL's FleetDNA project [7]. The subset of data used for this study contains a total of 8.6 billion samples from 1,718 commercial vehicles at a sample rate of one Hz. This dataset is unique in its size and breadth and forms the basis for multiple research endeavors into alternative fuel vehicle usage, alternative drive-train technologies, and emissions and fuel consumption of traditional combustion engines. While each vehicle provides many different data streams from multiple onboard sensors, this study considers only the wheel-based speed of each vehicle to support generalizability. The motivating goal here is to identify a typology of driving patterns in order to identify representative drive cycles for emissions testing [6].

Feature extraction is a key concern in the analysis of time series data. In particular, the ability to select relevant features for an analysis such as this arises from the expertise of vehicle researchers as well as by conventions in the domain. Only after relevant features are extracted is a metric space defined within which standard visualization and clustering techniques can be applied. Successful features allow for simpler models with better interpretability. One may prefer very powerful features so that multidimensional structure may be visualized and understood in a low dimensional space. On the other hand, simple features provide a direct connection between analysis and application domains. The interpretability of machine learning models can be crucial in certain applications, especially when transparency is desired [12].

The next section highlights prior work on methods for vehicle drive cycle segmentation and previous studies that have explored the efficacy and sensitivity of automatic time series clustering using other data sources. Section 3 describes the analysis methodology. Section 4 presents the results of the analysis. Section 5 discusses limitations of this work and opportunities for follow on work. Finally, section 6 concludes.

2 Background

2.1 Representative Drive Cycles in The FleetDNA Dataset

The Fleet DNA dataset contains millions of miles of historical drive cycle data captured at 1Hz resolution from medium and heavy vehicles operating across the United States. The dataset encompasses existing U.S. Department of Energy (DOE) activities as well as contributions from valued industry stakeholder participants. The Fleet DNA data used as a source for this analysis has been collected from a total of 30 unique fleets/data providers operating across 22 unique geographic locations spread across the United States. This includes locations with topology ranging from the foothills of Denver, Colorado to the flats of Miami, Florida.

The Environmental Protection Agency (EPA) sets emissions standards for a wide variety of vehicles which are each operated in a wide variety of ways. Refuse trucks, for example, may drive relatively slowly down a residential street, stopping at each drive-way. Long-haul semi trucks, on the other hand, may drive at higher speeds along interstate roads, stopping with much less frequency. The characteristics of each of these types of driving is captured in the drive cycle, which is a time series of vehicle speed versus time. The EPA is interested in identifying typical, or representative, drive cycles which can be used to create new emissions tests that are more representative of real world driving.

In prior work, the FleetDNA database was leveraged to identify key modalities of vehicle drive cycles in order to inform the development of representative drive cycles [6]. In the work, a set of hand-engineered features were chosen for analysis and ultimately eight key features were used for segmentation of vehicles. These eight features will be called "traditional" features in the rest of this paper and are listed in Table 1.

The traditional features are, for the most part, fully described by their names. Two of them, however, require a more detailed explanation. Characteristic acceleration represents the potential energy consumed by a vehicle as a result of acceleration and hill climbs. It is indicative of the maximum amount of energy that is potentially available for recapture via regenerative braking. Aerodynamic speed on the other hand is representative of the energy consumed by a vehicle over the course of a drive cycle via drag. This energy is lost to the environment and not able to be recaptured via regenerative braking. The ratio of Characteristic Acceleration to Aerodynamic speed *squared*, referred to as kinetic intensity, is helpful in identifying drive cycles which are ideally suited for hybridization/electrification. Higher KI values are indicative of drive cycles where a greater percentage of tractive energy can be recaptured, where low KI cycles have most of their energy being lost through drag.

Vehicle segmentation is performed using principal component analysis (PCA) for dimensionality reduction and k-medoids for clustering. Based on this analysis, three main clusters of vehicles were identified. The work is extended in this paper by consideration of a large number of additional automatic features.

2.2 Application Agnostic Features in TSFresh

TSFresh is a software package for Python which extracts characteristic features from time series data. According to the TSFresh website, the software has been used to successfully predict "the lifespan of machines" and "the quality of steel billets during a continuous casting process." Our purpose in this section is to provide a brief overview of TSFresh and its capabilities, without diving deeply into the details. We find that TSFresh has two main functionalities. The first is a feature extraction library which contains dozens of different feature extraction routines. These are python functions which take time series data and summarize them into one number. The second is a feature filtering, or feature selection, module which uses an array of significance tests to determine which of the extracted features are most important to predicting some labeling of the data. In our case, we wish to use the extracted features in an unsupervised fashion, producing clusters of optimal separation rather than labelings that most closely match some ground truth. We therefore leverage the feature extraction capability of TSFresh, but do not use the feature selection module. Table 2 offers a small sample of the features computed by TSFresh. The full set of features computed by TSFresh will be called "application agnostic" features in the rest of this paper.

Feature Name (units)	Description
1. Aerodynamic Speed (ft/s)	Describes the positive tractive energy required to overcome aerodynamic drag.
2. Characteristic Acceleration (ft/s^2)	Describes the positive tractive energy required to accelerate/climb a vehicle.
3. Percent Below 55 (%)	Percentage of distance accumulated at speeds below 55 mph.
4. Percent Zero (%)	Percentage of time duration accumulated at vehicle speeds of 0 mph.
5. Stops Per Mile	Number of times the vehicle makes a complete stop per mile.
6. Average Speed (mph)	Mean nonzero driving speed.
7. Maximum Speed (mph)	Maximum nonzero driving speed.
8. Speed Standard Deviation (mph)	Standard deviation of nonzero driving speed.

Table 1. Set of "traditional" features which were used in previous work by the authors of this paper. These features are compared with "application agnostic" features in the current work.

Many of the application agnostic features provide the same or similar information as the traditional features. For example, the variance and standard deviation computed by TSFresh provide the same information as the traditional feature "standard deviation of speed". However, the application agnostic feature set includes many more features such as skew and kurtosis. In fact, the application agnostic feature set goes well beyond the traditional features in their scope. All of the traditional features are second order or below. In other words, traditional features are computed with only information from, at most, two adjacent points in time. The feature set provided by TSFresh, on the other hand, contains features such as autocorrelation at various time lag. This feature may capture the nonlinear relationships between the data points in time which can not be captured by the traditional features.

In the following sections we present one analysis, conducted on a large, real world dataset, to determine how effective these application agnostic features are in extracting structure from the dataset.

2.3 Clustering Techniques for Time Series Data

In this subsection, we will discuss key techniques and the dominant trends in time series segmentation including methods for normalization, dimensionality reduction and clustering.

Normalization is done to prevent certain features from overtaking others due to their relative scale. In feature scaling, the minimum and maximum values of a given feature are set to 0 and 1, respectively. Values which fall in between are scaled linearly. Normalization is important at multiple stages in the clustering process. For feature extraction, dimensionality reduction, and clustering, data with highly varying scales can lead to ill conditioned numerical optimization. Dissimilar scales of components in the feature space can also cause unintended bias in quality metrics such as the silhouette score [3].

Dimensionality reduction techniques are used to project high dimensional data sets into lower dimensions while removing the minimal amount of useful information. In this paper two dimensionality reduction techniques are considered, t-distributed stochastic neighbor embedding (t-SNE) [19] and Principal Component Analysis (PCA) [8]. PCA was chosen so that results of this work could be interpreted with respect to results in the previous work. t-SNE was chosen as a representative of a class of nonlinear dimensionality reduction algorithms based on iterative optimization.

Clustering algorithms are designed to assign labels to unlabeled data. Some algorithms such as k-means require a-priori knowledge of the number of labels into which data should be separated. These types of algorithms work

Feature Name (units)	Description
abs_energy	Returns the absolute energy of the time series which is the sum over the squared values
absolute_sum_of_changes	Returns the sum over the absolute value of consecutive changes in the series x
agg_autocorrelation	Calculates the value of an aggregation function, in this case autocorrelation
approximate_entropy	Implements a vectorized Approximate entropy algorithm.
ar_coefficient	This feature calculator fits the unconditional maximum likelihood of an autoregressive AR(k) process.
value_count	Count occurrences of value in time series x.
variance	Returns the variance of x
variance_larger_than_standard_deviation	Boolean variable denoting if the variance of x is greater than its standard deviation.

Table 2. A sampling of features extracted by the TSFresh feature extraction module. There are more than 500 features which are extracted by default. Please see the TSFresh documentation for an exhaustive list of all features. This table is a partial reproduction of the table in that documentation.

by producing an initial guess and then iteratively refining that guess until some optimality condition is met. Other algorithms work by first assigning each data point to their own clusters and then iteratively merging labels together. All clustering algorithms considered for this study require some distance metric to be defined between each data point. This distance metric can be as simple as the euclidean distance, or quite complex as dynamic time warping (DTW), which includes a method to align the time series [3].

A substantial amount of prior work has been published exploring different techniques for clustering with time series data. The literature shows no indication of a clear best choice algorithm. Bagnall et al. provide a survey of time series clustering methods including 18 recently proposed and two benchmark time series classification algorithms applied to a set of 47 time series datasets [1]. They find the reference algorithms hard, but not impossible, to best. More than half of the proposed algorithms can beat the reference algorithms on average, with the best result (an algorithm invented by the same author), achieving a 9% improvement over reference implementations. Other state of the art time series clustering algorithms include K-Shape, which use a custom “shape based” distance measure which utilizes the cross correlation [16].

From these survey papers, it is apparent that two classes of time series classification algorithms are dominant in the literature. The first class are so-called "distance based" methods which treat the time series as a very high dimensional object and compare time series directly to one another. These algorithms rely on a distance metric that is defined between each time series directly. Such a metric is not easy to define. Euclidean distance, for example, requires each time series to be of the same length and is highly susceptible to misalignment. State of the art distance metrics are meant to remedy these problems. A review of these distance metrics can be seen in a survey paper by Liao. [11]

The second class consist of clustering algorithms are “charactertistic based.” These utilize traditional clustering algorithms in a vector space of features which are extracted in a separate feature extraction step. A diagram of characteristic based clustering is presented in Figure 1. This approach is advocated by Hyndman et al. [20]. In that work, the characteristics described are statistical features extracted from the time series such as skewness, kurtosis, seasonality, and serial correlation. Hyndman argues this approach is superior to distance based clustering, as it is more robust to errors and noise. Hyndman uses hierarchical clustering methods to produce clusters in the feature space and reports cluster quality in a metric called PU, which is said to be the number of correct bifurcations in the final hierarchical clustering divided by the total number of bifurcations. Hyndman explores how classification accuracy changes with respect to the number of characteristic features considered. Other approaches in characteristic

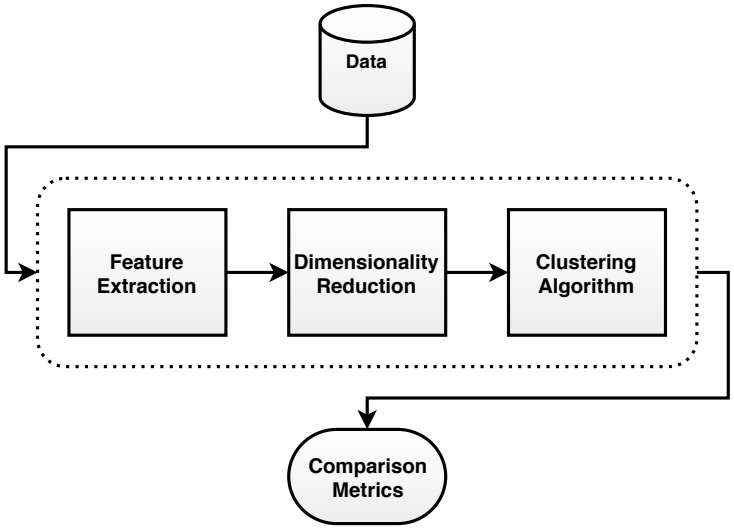


Figure 1. The generalized steps of a characteristic-based time series clustering algorithm as used in this paper. The three steps at the center of this chart: feature extraction, dimensionality reduction, and clustering algorithm, are the main components of the algorithm.

based clustering include the usage of the wavelet transform to extract more exotic features from the time series data [9].

Christ et al. extend the work of Hyndman by exploring a much larger set of features in the TSFresh software package for Python [5]. This software is used in their paper to feed a novel feature selection algorithm which accepts or rejects features based on a test of independence with respect to some prediction variable. Features which are statistically likely to have dependence on the prediction variable are said to be relevant. One potential issue with this approach is the tendency of the algorithm to select features which are highly correlated with one another. As a remedy, the authors suggest to perform analysis of the singular value decomposition, called PCAa(fter). This work utilizes the feature extraction machinery of this package, but does not utilize the feature selection algorithm. We are interested here in exploring the structure contained in the extracted features, without any notion of supervision.

Beyond these traditional approaches, there has been recent interest in applying deep learning architectures to the time series classification problem. Lankvist gives an excellent review of such techniques in [10]. The idea is to apply such models as restricted boltzman machines, autoencoders, and recurrent neural networks, to a related supervised learning problem. In a separate paper, Mandiraju et. al. use inter-neurons from an auto-encoder as features for a differentiable clustering algorithm based on t-SNE. The auto-encoder is trained simultaniously with the clustering algorithm [14]. One downside to using such powerful features is a lack of interpretability in domain-space. This work is focused on traditional approaches to automatic feature interpretation that have the key attributes of scalability and interpretability. Deep learning approaches including the auto encoder may be explored in future work.

3 Methods

In this paper, we explore the effect of swapping out the eight features used in the previous work on drive cycle segmentation with a set of domain agnostic features from TSFresh. To achieve this, we perform a sensitivity analysis using both the silhouette score and the variation of information observed in the clusterings produced using a grid search through various clustering algorithms. This method allows us to tease out the effect of swapping traditional, hand engineered, features for a set of automatically extracted features on those metrics. Subsection 3.1 describes the preprocessing steps applied to the FleetDNA dataset. Subsections 3.2 - 3.4 describe the different choices which can be made in the design of a clustering algorithm and choice in cluster quality metrics. Finally, subsection 3.5 describes how we use Apache Spark to tie all of these methods together in an exhaustive grid search through parameter space.

3.1 Data Preparation

Data was converted from an internal JSON (Javascript Object Notation) file format into CSV (Comma Separated Value) files, partitioned by vehicle. These CSVs were then combined into a one partitioned Parquet file for further analysis [1]. Some very basic quality control was performed on the data at this stage, such as looking at the first several rows of data for a sampling of vehicles to ensure the data looks correct. Separately from this study, the dataset has undergone physics based filtering to interpolate data containing physically impossible characteristics, such as instantaneous deceleration. More information on the quality control and filtering process can be viewed on the FleetDNA website. [7].

Next, The Apache Spark framework was used to segment each vehicle's time series data into "trips," which are separated by at least one minute of missing data. In this phase we also convert columns into the correct Spark data type, and then repartition the dataset by vehicle and sort by timestamp [2]. One interesting detail of implementation cropped up in this process, related to the definition of a trip. Since the trips may not be fully contained in any one window of time (such as a day, or a week), there is no easy way to express the trip labeling function in terms of Apache Spark window functions, which must collect each partition to a single executor node. To get around this issue, a custom lag function was implemented using the RDD API which does not require a partition to be specified.

3.2 Feature Extraction

After these preprocessing steps were completed, the Spark framework was used again to execute a UDF (User Defined Function) across all trips for each vehicle. This architecture allows us to perform computationally intensive feature extraction algorithms across each vehicle trip in the dataset, parallelized by vehicle trip. Due to the limitations of the automatic feature extraction package, TSFresh, we discard any trip with more than 50,000 data points for the computation of application agnostic features. We do this because TSFresh is not designed to handle very long time series data. We also chose to restrict the set of features to those tagged as "efficient" by TSFresh, since features with higher computational complexity were taking too long to run.

After extracting the traditional and agnostic features for each trip, these are aggregated by vehicle using a simple mean to obtain an average value. A simple mean was chosen to ensure that those features from shorter trips weighted equally to those from longer trips. Min-max scaling is then applied to both traditional and domain agnostic feature sets to guard against issues of magnitude.

Decorrelation between features is performed on the domain agnostic features only. This was implemented after exploratory analysis revealed highly skewed cluster in the PCA projection. Upon inspection, many of the application agnostic features turned out to be highly correlated to one another (a problem that is discussed by the authors of TSFresh in [5]). We remove correlated features by first sorting the features alphabetically. Next, we loop through

[1] Alenander Van Roijen, a SULI intern, performed these steps using NREL's Peregrine supercomputer.

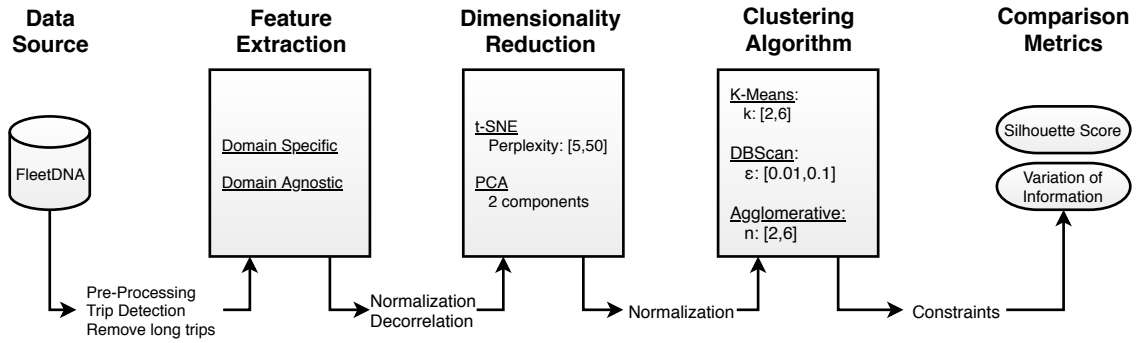


Figure 2. Flowchart of the segmentation process with hyper-parameters displayed for each decision point in the analysis. All computation paths are taken and the results aggregated by metric.

each feature and compute the correlation coefficient *overvehicles* with all other features. Any feature which has a correlation coefficient greater than 0.9 to any other feature is deleted. The first feature *lowerinlexicographicalindex* of any correlated pair is kept. This process is then repeated until no correlated features remain. In total, 431 of 723 features are kept.

The application agnostic feature extraction is implemented in a parallelized fashion using the Spark cluster computing framework. The code, as well as the output data, is available from our github repository.

3.3 Dimensionality Reduction and Clustering Algorithm

Dimensionality reduction was done using both PCA and t-SNE. The PCA algorithm produces a linear map of input features down to a reduced dimensionality by determining the subset of components (vectors) that describe the greatest amount of variation in the underlying data. The t-SNE algorithm is a more recent one and uses the gradient descent algorithm to produce a nonlinear mapping of input features into a reduced space. This algorithm is sensitive to its parameters, such as the perplexity, and is known for its ability to produce meaningless clusters when the perplexity parameter is not set correctly. This is undesirable in most cases, as the clusters come from noise in the dataset and not necessarily an underlying structure. Perplexities for this analysis are considered in the range recommended by Van Der Maaten [19].

Three different clustering algorithms: k-means, DBSCAN, and Agglomerative Clustering are considered. The k-means algorithm was chosen for consistency with previous work. The DBSCAN and Agglomerative Clustering algorithms were chosen for their good performance, ability to cluster varied types of data, and because they are representative of classes of clustering algorithms. DBSCAN represents a class of density based algorithms which separate data into clusters by comparing the density of data points to a threshold density. Agglomerative clustering represents a class of hierarchical clustering algorithms in which data points are iteratively merged together into clusters based on some distance metric. The implementations of all these algorithms can be found in the Python package scikit-learn [17].

3.4 Comparison Metrics

Several metrics of quality are computed from each clustering so that comparisons may be drawn between methods. The quality metrics chosen for this analysis are silhouette score and variation of information with respect to three baseline labels: vehicle class, type, and vocation.

The silhouette score is a measure of the compactness of each cluster, and the cleanness of separation between clusters. For one cluster, it is defined as the ratio of the average distance between all points within a given cluster and

Algorithm	Parameters
t-SNE	10 samples of Perplexity $\in [5,50]$
PCA	n/a
k-means	k $\in \{2,3,4,5\}$
DBSCAN	10 samples of $\varepsilon \in [0.01,0.1]$
Agglomerative	$n \in \{2,3,4,5\}$

Table 3. An enumeration of hyper-parameters for each step in the segmentation algorithm. Each combination was used to compute quality metrics of labelings using each feature set.

those points outside the given cluster. The silhouette score for the whole dataset is the average silhouette score of every cluster within the dataset [18].

Variation of information is a measure of the mutual information between two clusterings. The metric is only meaningful with respect to a baseline clustering, and it is only the ordering of the metric (not necessarily the magnitude) which can be readily interpreted [13]. These baselines are taken from the metadata of the FleetDNA dataset. Vehicle class is a numeric label which categorizes vehicles by weight. There are eight classes of vehicles. For example: Class 1 represents vehicles 0lbs to 6,000 lbs, while Class 8 represents vehicles from 33,001lbs to 100,000lbs. Vehicle type is a categorical label which represents the vehicle's manufacturing class. Labels include: SUV, Tractor, Minivan, Dump, and Fire Truck. Vehicle vocation is a categorical label which represents the way in which the vehicle is being utilized by the driver. Examples of these labels include Long Haul, Mass Transit, Snow Plow, Parcel Delivery, and Refuse Pickup [7].

3.5 Grid Search and Aggregation

We now have an analysis involving two sets of features, two dimensionality reduction techniques (with one tuning parameter considered overall), as well as three clustering algorithms (with three tuning parameters considered overall). From these analysis, four comparison metrics are computed: sillhouette score, and variation of information of class, vocation, and vehicle type. We once again utilize the Apache Spark framework to perform this computation in a distributed fashion. The implementation leverages the "embarrassingly parallel" nature of this grid search, in that each combination of experimental parameters can be sent to a different processing node.

To determine the sensitivity of comparison metrics to the feature set, results from the perspective of two different analyses are presented. First, we vary the number of features selected from the domain agnostic feature set to calculate of the sensitivity of clustering to those features. Second, the factors in each segmentation model are varied to isolate the effects of each algorithmic choice. In this way, factors contributing to the quality metrics are isolated.

For the first analysis, the number of domain agnostic features is varied. Silhouette score is plotted as the number of agnostic features is increased. The number of features is $n \in [8,18,...378]$, and we take sample, with replacement, over 25 trials per value of n. k-means is then performed with 3 centroids. In total 3,801 experiments are performed. Of those, 12 experiments were filtered out due to failing the constraints. These ensure that each clustering contains at least three clusters and that no cluster contains less than 10 vehicles.

For the second analysis, a grid search is performed over each combination of segmentation algorithms. The parameters searched over in each section of the segmentation algorithm are shown in Table 3. A flowchart outlining the process is shown in figure 2. Features are chosen to be either the traditional or agnostic feature set. Precisely 50% of trials take advantage of each feature set. Dimensionality reduction is then applied. PCA and ten different configurations for t-SNE (perplexities sampled linearly between 5 and 50) are considered. This creates a bias towards t-SNE, since there are more samples from this method. The effectiveness of t-SNE to the effectiveness of PCA are not being compared directly, so this bias is acceptable. Clustering is then performed using k-means with k varying between 2 and 5, DBSCAN with 10 samples of epsilon between 0.01 and 0.10. The implementation of these algorithms is taken

from the open source Python package scikit-learn [17]. Enumerating through all of these options gives us a total of 396 experiments to evaluate.

4 Results

Visualization of vehicles in both feature spaces is achieved using a PCA scatter plot and colored by k-means (k=3). This visualization is shown in Figure 3. The domain agnostic features produce a scatter plot with multiple well defined clusters. This scatter plot seems to have more structure than the scatter plot produced by domain specific features, which has one main blob and two smaller clusters on the top and bottom. Both scatter plots provide a main cluster with the majority of vehicles, and several clusters of vehicles on the periphery.

Figures 4 and 5 show the results of computing the silhouette score as two different parameters are varied. It is shown that the number of clusters chosen for k-means has a large impact on the silhouette score. The results confirm what can be seen visually in the PCA plot of Figure 3. Domain specific features identify two main clusters while domain agnostic features identify three main clusters.

Varying the number of domain agnostic features in Figure 5 yields transient behavior for models with less than 30 input features. This transient does settle to a mean silhouette score comparable with the optimal score from the domain specific features after 100 features are included.

The results of the sensitivity analysis are visualized in Figure 7 and summarized in Table 4. The distributions of silhouette score are broken down by factor in the segmentation model, and then further broken down by feature set. A mean silhouette score of approximately 0.4 can be observed across all methods and feature sets, with some variation between the methods. DBSCAN, in particular, seems to produce lower silhouette scores than other methods.

One interesting effect is the generally higher means of the silhouette scores of agnostic features within each bin. This is a quantitative indicator that using domain agnostic features results in clusters with better separation, no matter which clustering algorithm is used. This effect is most pronounced when using the t-SNE algorithm, where a clear bimodal distribution of silhouette scores can be seen.

Variation of information was not significantly affected by any of the factors tested. One possible exception is a tendency of traditional features to produce low variation of information scores when using traditional features. On the other hand, a slightly lower variation of information is observed with agnostic features when using the k-means dimensionality reduction or the agglomerative clustering algorithm. The results of these analysis are inconclusive, but the variation of information with vehicle class, vehicle type, and vehicle vocation are not significantly affected by the feature set.

This result is interesting for two reasons. Firstly, the eight hand engineered domain specific features were able to extract a similar amount of information from these time series (with respect to the metadata considered) as were over three hundred domain agnostic features which took much longer to compute. Secondly, the domain agnostic features drawn at random out of a pre-made statistical package were able to extract a similar amount of information as the hand engineered features created from expert knowledge. This indicates that both benefits and drawbacks are to be expected when using domain agnostic features.

Centroids of all segmentation models from domain agnostic features are visualized in a density plot in Figure 6. The axes of the joint histogram are two features from the domain specific feature set: Percent zero speed and average driving speed. The centroids obtained from domain agnostic features were therefore reprojected into this subspace of domain specific features. The centroids from previous study [6] are included as red crosses over this figure. Two of the cluster centroids from previous analysis seem to overlap the two main regions of dense agnostic cluster centroids.

This figure reveals a dense collection of cluster centroids around the middle of the plot. Three arms can be observes leading away from the center. One arm leads towards high speed vehicles with an average percentage of zero speed. The other two tendrils head in opposite directions along the percent zero axis, indicating one group of vehicles that are largely idle, and one group of vehicles that are rarely idle.

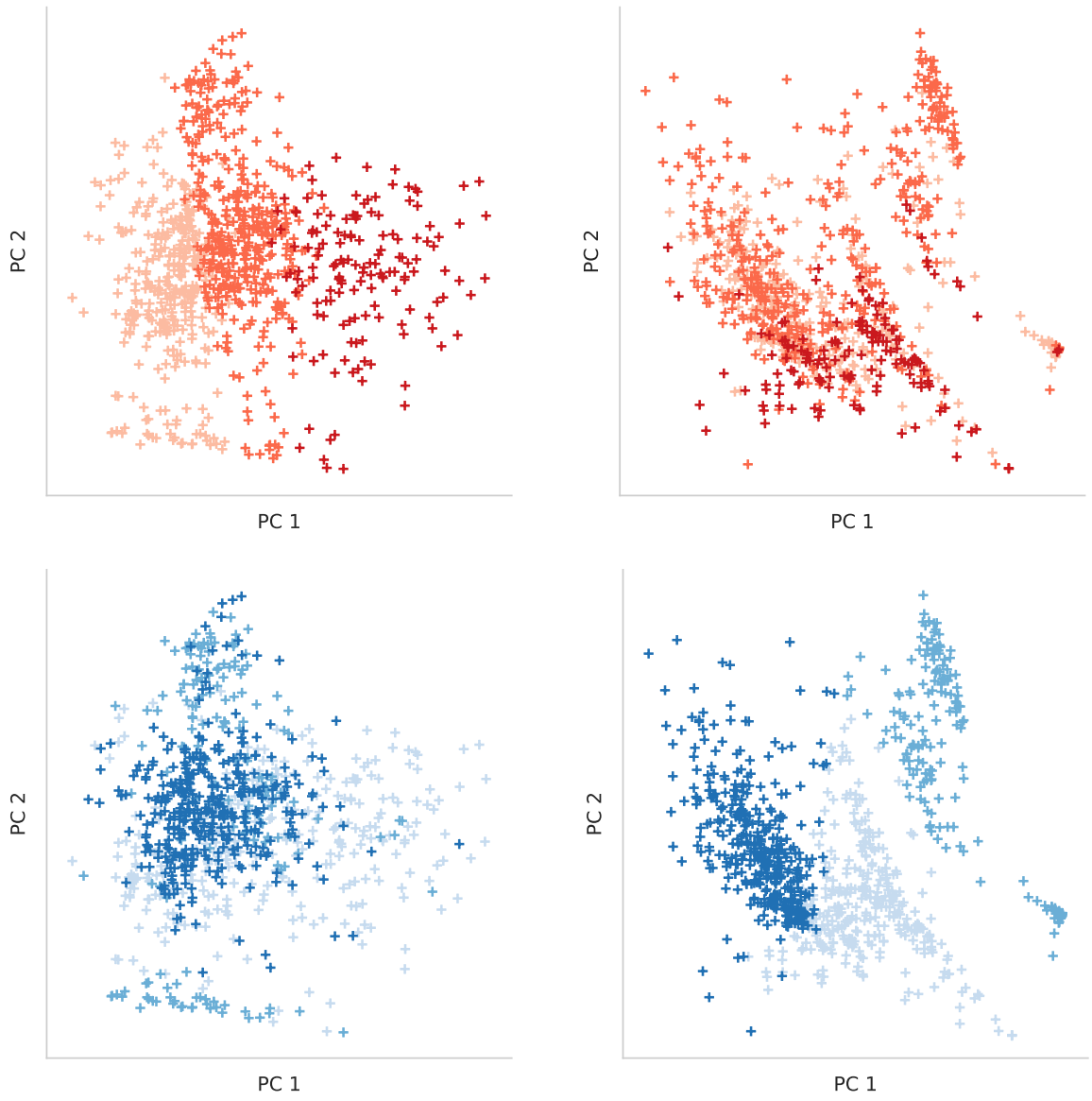


Figure 3. PCA plots colored by k-means algorithm with (n=3) using both domain specific and domain agnostic features. The top row (red) is shaded by clusters derived from domain specific features. The bottom row (blue) is shaded by clusters from domain agnostic features. The left column is projected using domain specific features, while the right column is projected using domain agnostic features. This tableau reveals some structure in both feature sets, and how labels are shuffled around when data is transformed between spaces.

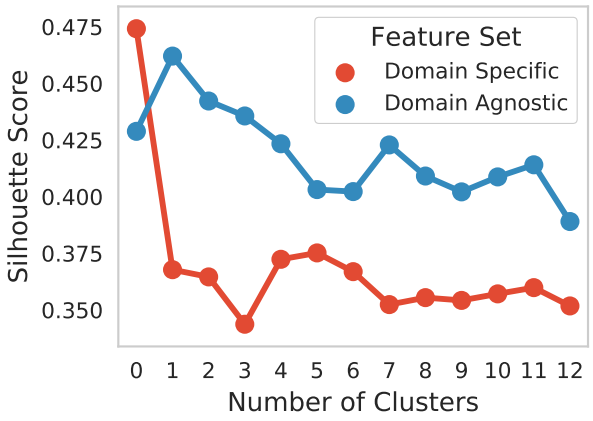


Figure 4. Silhouette score for domain agnostic and domain specific features under k-means clustering as number of clusters is varied. The domain specific feature set seemingly maximizes the silhouette score with two clusters, while the domain agnostic feature set maximizes the silhouette score with three.

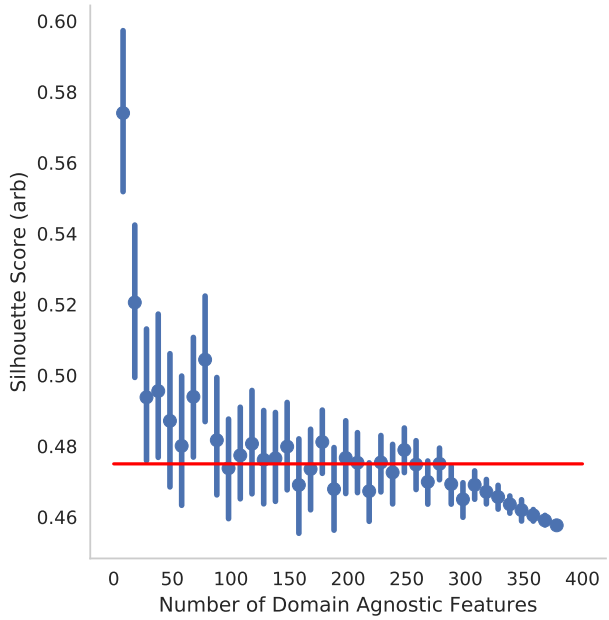
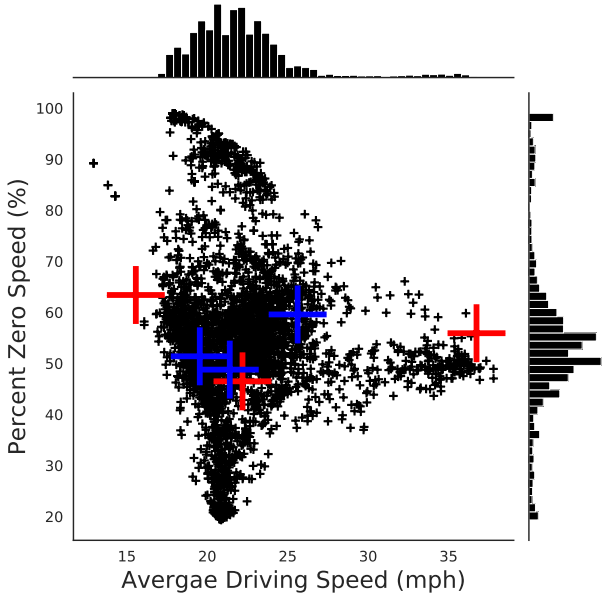


Figure 5. The silhouette score under k-means with k=3 as number of agnostic features is varied. Red line is the silhouette score achieved by using the eight domain specific features. The sillhouette score is higher than average before 100 features are included. This settles around the same score reported from domain specific features, followed by a sharp decrease in silhouette score above 300 features.



1

Figure 6. Density of centroids for k-means on a random sampling of 50 to 60 domain agnostic features. The density is plotted on two features from the domain specific feature set. Centroids of clusters from the domain specific features are shown as red crosses.

Metric:	Silhouette Score		VI Vehicle Type	
Feature:	Agnostic	Specific	Agnostic	Specific
DBSCAN	0.52 (0.09)	0.24 (0.09)	3.36 (0.27)	3.11 (0.21)
KMeans	**0.54 (0.07)**	0.44 (0.03)	3.54 (0.15)	3.48 (0.17)
PCA	0.44 (0.02)	0.38 (0.02)	3.45 (0.10)	3.44 (0.02)
TSNE	0.53 (0.09)	0.33 (0.12)	3.40 (0.26)	3.27 (0.27)

Metric:	VI Class		VI Vocation	
Feature:	Agnostic	Specific	Agnostic	Specific
DBSCAN	2.79 (0.33)	2.53 (0.33)	4.14 (0.17)	3.92 (0.21)
KMeans	3.04 (0.17)	3.12 (0.15)	4.18 (0.14)	4.07 (0.17)
PCA	2.92 (0.11)	3.05 (0.10)	4.19 (0.21)	4.06 (0.06)
TSNE	2.86 (0.32)	2.79 (0.40)	4.15 (0.16)	3.98 (0.21)

Table 4. Results for each comparison metric in the sensitivity analysis. Standard deviation is given in parenthesis. Silhouette scores generated from domain agnostic features are generally higher than those generated from domain specific features. The two values highlighted in bold are silhouette scores for domain agnostic features which appear significantly higher. Variation of information scores do not change significantly with feature set. This table summarizes that which is visualized in Figure 7

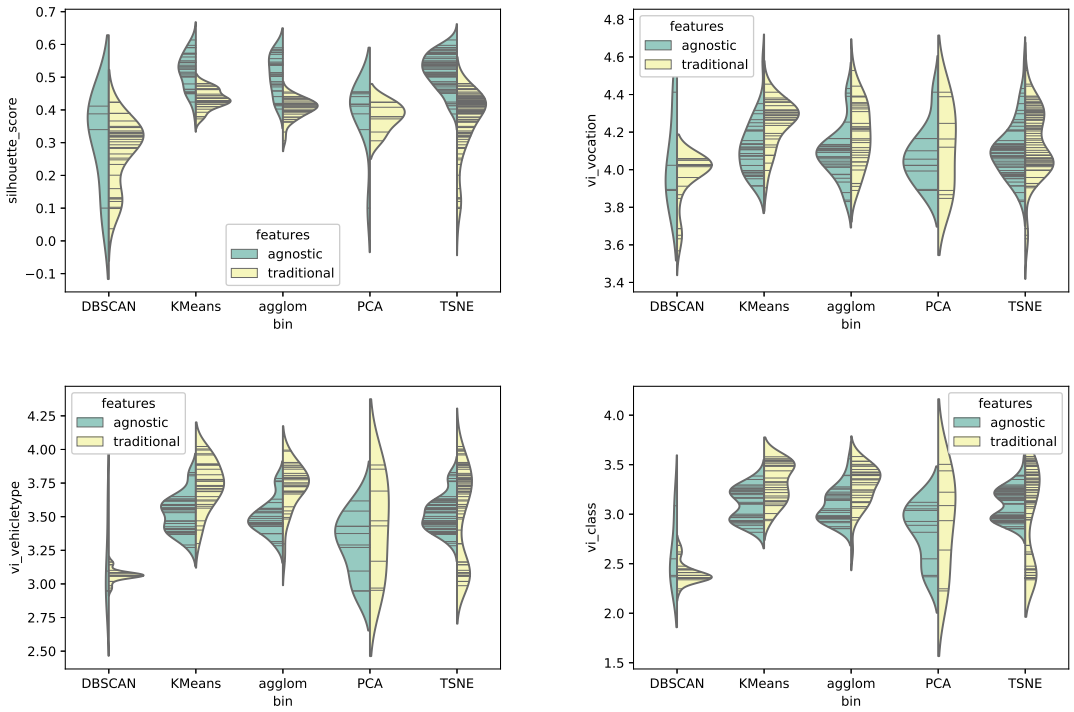


Figure 7. Distribution of silhouette scores by factor in the segmentation model. Each bin on the horizontal axis represents a factor in the segmentation model which is held constant. The vertical axis is the quality metric: Silhouette score or Variation of Information. Each bin is split into feature set. Data points contained in the green distributions were created from domain agnostic features, while data points contained in the yellow distributions were created from domain specific features. This figure visualizes that which is summarized in Table 4

5 Discussion

The results of this analysis are encouraging, as they lends evidence that similar (if not more) information is contained in the domain agnostic feature set as is contained in the domain specific feature set. In some metrics, such as silhouette score, better quality is observed by using domain agnostic features. Other metrics, such as variation of information, show statistically insignificant changes between feature sets.

Visual inspection of data via PCA reveals different structures in each space of features. For domain agnostic features, five overlapping clusters are visible, each with a slightly stretched appearance - hinting at correlations between features. For the domain specific features, one large main cluster and a line of outliers at the bottom was observed. This was found in previous work to be a cluster of school buses. Using a new set of domain agnostic features uncovers a competitive amount of structure in the FleetDNA dataset when compared to domain specific features.

Some pre-processing steps should be explored more thoroughly in future work. The definition of a trip as more than five minutes of missing data and to throw out any trip with more than 50,000 data points was driven by both expert opinion and out of necessity, as the TSFresh software package could calculate features for very long time series. The effect of other pre-processing routines, and alternate data partition schemes, which are not not detailed in this paper could also be explored in future work.

Feature selection was limited to those marked as “efficient” in the TSFresh software package to speed up computation time. Future work could be done to explore more types of domain agnostic features and perform sensitivity analysis to the specific subtypes. A more fine grained approach to feature selection (than random sampling) would yield a more thorough analysis.

The method of averages used to aggregate trip-level features into vehicle-level features was driven by convenience. Some features, such as percentile, should not simply be averaged to form an aggregate statistic. This design choice was dictated by the design of the TSFresh library, which does not easily support parallelism at the time step level. In future work, agnostic features with time step level parallelism may be considered.

Another potential concern is that inclusion of so many domain agnostic features runs the risk over overfitting. It seems quite natural that more features should be able to contain more information about the underlying data. This is an interesting question to consider in the future. Similar scores are achieved when considering just eight hand engineered features as over three hundred domain agnostic features.

There are potential hidden biases in the sensitivity analysis. For example, data is normalized before metric extraction to try and control for the property of scale, but there may be other factors which are not considered and would lead to biased results in some cases.

Lastly, the interpretability of these and other domain agnostic features is opportune for future study. Additional work is also needed to provide a thorough comparison between domain agnostic features and those features extracted through deep learning.

6 Conclusion

This work has shown that a set of domain agnostic features taken from the TSFresh library can be used to produce clusters of time series which are comparable (if not slightly superior) in silhouette score to eight hand engineered features. Furthermore, this finding is robust for at least two dimensionality reduction techniques (PCA and t-SNE) and three clustering techniques (k-means, agglomerative, and DBSCAN).

There is evidence that the contents of the discovered clusters is likewise similar between the two feature sets. A similar level of variation of information (with respect to three labels in the metadata) is found between both feature representations. This similarity is striking, since it means similar information is present in the domain agnostic features as is present in the domain specific features. It does so, however, using two orders of magnitude more features.

Thus, characteristic based clustering using domain agnostic features may be a useful tool for exploratory data analysis when hand-engineered features by domain experts are unavailable. This method may also be useful in the discovery of features for application domains where none are widely agreed upon in the literature.

Bibliography

[1] et al Anthony Bagnall. "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances". In: *Data Min Knowl Disc* (2017).

[2] Apache Software Foundation. *Apache Spark Homepage*. https://spark.apache.org/. 2018.

[3] Donald J. Berndt and James Clifford. "Using Dynamic Time Warping to Find Patterns in Time Series". In: *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*. AAAIWS'94. Seattle, WA: AAAI Press, 1994, pp. 359–370. URL: http://dl.acm.org/citation.cfm?id=3000850.3000887.

[4] Bruce Bugbee et al. "Prediction and characterization of application power use in a highâĂŘperformance computing environment". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 10.3 (2017), pp. 155–165. DOI: 10.1002/sam.11339. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sam.11339. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/sam.11339.

[5] Maximilian Christ, Andreas W. Kempa-Liehr, and Michael Feindt. "Distributed and parallel time series feature extraction for industrial big data applications". In: (2016). arXiv: 1610.07717. URL: http://arxiv.org/abs/1610.07717.

[6] Adam Duran et al. "Leveraging Big Data Analysis Techniques for U.S. Vocational Vehicle Drive Cycle Characterization, Segmentation, and Development". In: *WCX World Congress Experience*. SAE International, 2018. DOI: https://doi.org/10.4271/2018-01-1199. URL: https://doi.org/10.4271/2018-01-1199.

[7] "Fleet DNA Project Data". In: *National Renewable Energy Laboratory* (2017). URL: www.nrel.gov/fleetdna.

[8] Karl Pearson F.R.S. "LIII. On lines and planes of closest fit to systems of points in space". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572. DOI: 10.1080/14786440109462720. eprint: https://doi.org/10.1080/14786440109462720. URL: https://doi.org/10.1080/14786440109462720.

[9] et al Hai Zhang. "Multiscale Feature Extraction for Time Series Classification with Hybrid Feature Selection". In: *ICIC 2006* (2006).

[10] Martin Längkvist, Lars Karlsson, and Amy Loutfi. "A review of unsupervised feature learning and deep learning for time-series modeling". In: *Pattern Recognition Letters* 42.1 (2014), pp. 11–24. ISSN: 01678655. DOI: 10.1016/j.patrec.2014.01.008. arXiv: 1602.07261.

[11] T. Warren Liao. "Clustering of time series dataâĂŤa survey". In: *Pattern Recognition* 38.11 (2005), pp. 1857 –1874. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2005.01.025. URL: http://www.sciencedirect.com/science/article/pii/S0031320305001305.

[12] Zachary Chase Lipton. "The Mythos of Model Interpretability". In: *CoRR* abs/1606.03490 (2016). arXiv: 1606.03490. URL: http://arxiv.org/abs/1606.03490.

[13] Marina MeilÇŐ. "Comparing clusterings-an information based distance". In: *Journal of Multivariate Analysis* 98.5 (2007), pp. 873–895. ISSN: 0047259X. DOI: 10.1016/j.jmva.2006.11.013.

[14] Dimitry Fisher Homa Karimabadi Naveen Sai Madiraju Seid M. Sadat. "Deep Temporal Clustering: Fully Unsupervised Learning of Time-Domain Features". In: (2018).

[15] Rafael Orozco, Shuangwen Sheng, and Caleb Phillips. "Diagnostic Models for Wind Turbine Gearbox Components Using SCADA Time Series Data". In: *IEEE PHM* (2018).

[16] John Paparrizos and Luis Gravano. "k-Shape: Efficient and Accurate Clustering of Time Series". In: *SIGMOD Rec.* 45.1 (June 2016), pp. 69–76. ISSN: 0163-5808. DOI: 10.1145/2949741.2949758. URL: http://doi.acm.org/10.1145/2949741.2949758.

[17] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[18] Peter J. Rousseeuw. "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis". In: *Journal of Computational and Applied Mathematics* 20 (1987), pp. 53 –65. ISSN: 0377-0427. DOI: https:

//doi.org/10.1016/0377-0427(87)90125-7. URL: http://www.sciencedirect.com/science/article/pii/0377042787901257.

[19] L J P Van Der Maaten and G E Hinton. "Visualizing high-dimensional data using t-sne". In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605. ISSN: 1532-4435. DOI: 10.1007/s10479-011-0841-3. arXiv: 1307.1662. URL: https://lvdmaaten.github.io/publications/papers/JMLR{_}2008.pdf{\%}0Ahttp://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=pubmed{\&}cmd=Retrieve{\&}dopt=AbstractPlus{\&}list{_}uids=19114314791487345 48related:VOiAgwMNy20J.

[20] Rob Hyndman Xiaoze Wang Kate Smith. "Characteristic-Based Clustering for Time Series Data". In: (2006).