Datenbank Spektrum (2021) 21:237–244
https://doi.org/10.1007/s13222-021-00391-x

FACHBEITRAG

Feature Engineering Techniques and Spatio-Temporal Data Processing

Chris-Marian Forke[1] · **Marina Tropmann-Frick**[1]

Received: 31 May 2021 / Accepted: 26 September 2021 / Published online: 20 October 2021

Abstract
More and more applications nowadays use spatio-temporal data for different purposes. In order to be processed and used efficiently, this unique type of data requires special handling. This paper summarizes methods and approaches for feature selection of spatio-temporal data and machine learning algorithms for spatio-temporal data engineering. Furthermore, it highlights relevant work in specific domains. The range of possible approaches for data processing is quite wide. However, in order to use these approaches with the spatio-temporal data in a meaningful and practical way, individual data processing steps need to be adapted. One of the most important steps is feature engineering.

Keywords Feature engineering · Feature selection · Spatio-temporal data

1 Introduction

These days, spatio-temporal data are widely used and common. Therefore, it is often required to process them using machine learning algorithms. Since this is a complex task due to the data's unique structure with the focus on both, time and space, the research in this area continues. In comparison, usually machine learning is based on loose feature records which have no relation with each other. If they do, then only in one dimension, such as time. This makes spatio-temporal data especially interesting. In this work we summarize the most common methods which are currently applied in different domains. Based on categorization done in [2] and [23], we indicate which methods are suitable for which kind of data and how these methods work in general.

Before we can dive into the data processing part, it is important to take the preprocessing steps into account, especially the feature engineering part. There are several steps in the feature engineering process. One of them is feature selection. This is considered in more detail in this paper. It involves the selection of the most relevant features from the total set, which can be used to train more efficient models than is the case with all features. The reason for this might be that certain features are duplicated, irrelevant, or otherwise confuse the machine learning method used.

Feature Selection is often associated with different phases of the feature engineering process. A clear representation of the process is therefore very important.

First, new features are extracted from the existing ones during feature extraction. This step can also involve other data sources. That happens in order to extract relevant details from the features and make them more recognizable. In addition, enriching existing data with foreign data sources can provide further information that cannot be detected by an algorithm using existing data, such as whether a day is a holiday for example.

The most important features are then selected in the next step during feature selection. This means that existing features are deleted if they are irrelevant. In other words, we want to find the most meaningful subset of features that is available in our dataset. Features may be dropped because they are either redundant with other features, simply irrelevant to the result, never change and thus have no effect, are duplicated, or strongly correlate with other features.

In Chapt. 2 we focus on the issues of feature selection in general. Chapt. 3 deals with different forms of spatio-temporal data occurrence. Also, a mapping to possible data representations is shown for each of the forms. In Chapt. 4, the corresponding machine learning methods are listed. We indicate also here, when should which method be applied for. Subsequently, in Chapt. 5, the most important data mining tasks are explored and some examples are given, how they can be implemented using which machine learning

✉ Marina Tropmann-Frick
marina.tropmann-frick@haw-hamburg.de

[1] Hochschule für Angewandte Wissenschaften Hamburg, Berliner Tor 7, 20099 Hamburg, Germany

methods. We conclude this paper with a summary and an overview over future work.

2 Feature Selection

Feature selection can be performed in different ways. Mainly, these methods can be divided into three groups: filter methods, wrapper methods and embedded feature selection methods. [3, 10].

Filter and wrapper methods evaluate all subsets of the features. This allows the subset with the best score to be selected. Filter methods do this by using special metrics that evaluate individual features or compare multiple features with each other.

Wrapper methods, on the other hand, achieve this by training one model for each of the different subsets. Thus, the subset whose model has achieved the best accuracy can be selected. Embedded feature selection methods integrate feature selection into the training process. Thus, they dynamically remove features that have little or no relevance due to the trained weights.

2.1 Filter Methods

Filter methods [18] are the simplest of the three types, as they use simple metrics to score each feature. Then the highest scoring features can be selected and used to train the model. This simplicity also makes these methods very fast.

2.1.1 Univariate and Multivariate Filter Methods

Filter methods can be distinguished between univariate and multivariate methods [12]. Univariate methods refer to individual features and evaluate each of them independently of the others. This is used, for example, to find invariant features or features that have no impact on the result.

Multivariate methods, on the other hand, look for correlations between features. They can be used to find redundant, duplicate, or correlated features.

2.1.2 Basic Filter Methods

Basic filtering methods include, for example, searching for features that are always the same or almost always the same, or finding duplicate features. Filtering out such features is a rudimentary and very fast method that greatly simplifies the entire feature engineering process.

2.1.3 Correlation Filter Methods

The correlation filter methods belong to the multivariate methods. These are methods that detect and filter out correlations between two features. If two features are correlated, then it is sufficient to keep one of them, since they carry the same information and can make the model unnecessarily complex.

There are different ways of calculating correlations, described in [1]. Pearson's correlation coefficient is commonly used, it calculates the linear correlation. Spearman's and Kendall's rank correlation coefficients, on the other hand, check whether the progression is monotonic, i.e. whether the order is preserved.

2.1.4 Statistical and Ranking Filter Methods

The last type of filter methods contains functions that are partially very similar to the correlation filter methods. This includes the check whether features contain mutual information [22], i.e. it is possible to infer from one to the other. Differently than with the correlation filter methods this is computed however in relation to the target value. Therefore, it provides information about how much a feature reveals directly about the label.

Another calculation that has the same goal is the chi-square evaluation, as seen in [21]. However, it only applies to categorical features.

A further method is the construction of decision trees based on exactly one feature only. The different decision trees are then compared and the features that provide higher accuracy are ranked higher. For example [16] describes this as well in a more general approach.

2.2 Wrapper Methods

The next method group is that of wrapper methods [3, 7]. In contrast to the filter methods, the different feature subsets are evaluated here rather than features individually. This takes into account that sometimes features have to be combined to predict the label. To achieve this evaluation, a model is trained with the feature subset in each individual case and its loss is used as an evaluation for the corresponding subset. This approach has the disadvantage that it depends on the particular machine learning algorithm. On the other hand, a correspondingly better result can be achieved. However, since one model must be trained for each subset, the computational effort increases enormously.

Such an approach consists of two to three components that can be combined in an arbitrary way. On the one hand, a search strategy is required that determines the feature subsets to be tested, and on the other hand, a learning algorithm is selected. In addition, a stop criterion is needed if not all combinations are to be tested. A stop criterion could be, for example, that a certain number of features has been reached or that the accuracy of the model no longer shows any significant improvements.

2.2.1 Search Strategies

There are many different search strategies. [8] describes many of them. Here is a short summary.

Forward and *Backward* Feature Selection are the simplest search strategies. The first one starts with no features at all and then adds the most promising ones. The latter starts with all features and removes one at a time.

Since both have the problem that later an earlier step may become unnecessary, there is Plus-L, Minus-R as an alternative, where alternately L features are added and then R features are removed. An extension of this is *Sequential Floating*, where L and R are calculated dynamically.

Forward and Backward Feature Selection can also be combined to the *Bidirectional* search strategy. Then both are executed in parallel; however, Forward Feature Selection does not add anything that Backward Feature Selection has already sorted out, and vice versa.

Moreover, there is *Exhaustive* Feature Selection at which all possible subsets are processed. This variant takes the longest, but also finds an optimal solution. Furthermore, no stop criterion is needed for this strategy, since the program tests every possible subsets.

2.3 Embedded Feature Selection Methods

Embedded feature selection methods do not select features until the training process [3]. Like the wrapper methods, these methods consider that features sometimes complement each other to predict the label. However, they are very fast, similar to the filter methods.

The approach is to derive the relevance of the features while training the model. The features that are classified as less important are then removed accordingly. Depending on the model, there are two approaches. In the case of a decision tree, this can be done directly, otherwise the relevance calculation takes place via regularization.

2.3.1 Regularization

The $L1$ regularization punishes features so that overfitting does not occur. A regression coefficient can drop to 0, which means that the feature can be ignored. In this case, it can be removed from the model since it no longer has any influence. This is also possible when combining $L1$- and $L2$-regularization [27], but not when using $L2$-regularization only, since the regression coefficients cannot become 0 then.

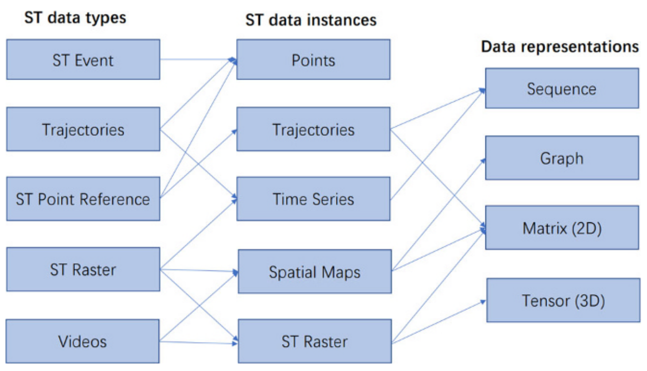

Fig. 1 Data instances and representations of different ST data types [23]

2.3.2 Decision Trees

In the case of decision trees, the relevance of a feature can be derived by how much it is used to split branches. If a feature is rarely used as a criterion, it is less important. In addition, the purity of the distribution is important. If the results still contain many records of different classes after the split, the feature is also less important. Thus, the less important features can then be removed [9].

3 Spatio-Temporal Data

Generally, spatio-temporal data (ST data) can be separated into five categories, as proposed by [2] and extended by [23]. Different data types are stored differently and represented differently again. These categories are considered more closely below. Fig. 1 shows their relations.

Depending on the domain, we distinguish between continuous and discrete spatio-temporal fields. Continuous fields have a value for every location at every time (such as temperatures). Discrete fields only have values at certain locations and certain times (e.g., crimes). They can be recorded in their entirety while continuous fields cannot.

3.1 Event Data

Event data are records of events occurring at certain places at certain times. Every discrete event is saved with its location and time, a so-called point. Formally, this is represented by a tuple (e, l, t) with e being the event itself, l being the location and t the time. This results in a very unstructured set. There are times with multiple events and times without any events. Also, events can occur everywhere. An example for a set of event data can be seen in Fig. 2.

We can explore this kind of data mostly when tracking something, e.g., crimes, accidents, etc. Generally, these events are loose. Often however, there are connections be-

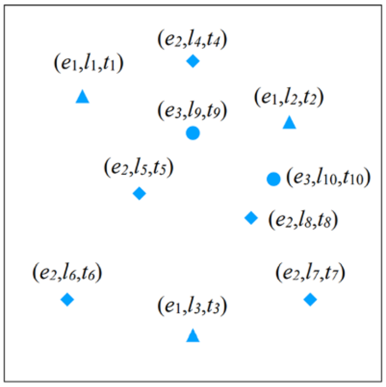

Fig. 2 Example for a set of event data [23]. Three types of events are shown here (e_1, e_2, e_3), which happened at different locations (l_x) and at different time points (t_y)

tween these events. For example, some regions may have a higher criminality than others do.

3.2 Trajectory Data

Trajectories are paths of objects. Objects can be anything, like people riding bikes, or balloons. Trajectories can be recorded e.g. by GPS.

Since they are continuous routes, they cannot be saved with all the details, but merely with an approximation. This can either be certain discrete points including the object, where it is and at what time it is there for many points in time. How this looks like can be seen in Fig. 3. Or this can be a time series item which include the locations in chronological order for each specific object. In the latter case, they can be represented as sequences. Formally, such a sequence is $\{(l_1, t_1),(l_2, t_2),...,(l_n, t_n)\}$ with l_x being the location at step x and t_x being the time at step x.

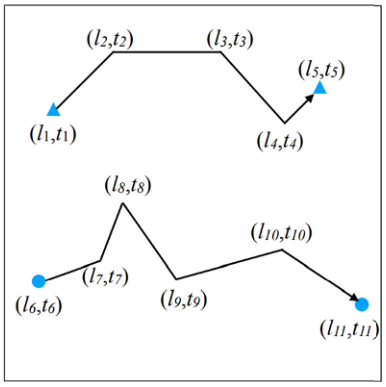

Fig. 3 Example for a set of trajectory data [23]. Two trajectories are shown here, with all points specified by a location (l_x) and a time (t_y)

3.3 Spatio-Temporal Point References

This kind of spatio-temporal data occurs when the domain is a continuous spatio-temporal field. A good example for this kind of data is the temperature. At all times, there is a certain temperature at all locations. However, since the temperature cannot constantly be measured at all locations, point references are recorded. In the given example, this might be done by weather balloons. They measure the temperature at different locations at different times. An example for such a measuring is shown in Fig. 4.

As a result, the data consists of temperatures for different places with individual timestamps. Out of these maps that look different at every moment, the continuous field can be calculated using domain specific algorithms.

To store the data, either every measurement can be saved as a point with its time and place. Then, the result is a set of tuples exactly like those of event data (which was (e, l, t) with e being the measurement, l being the location (dimension depends on use-case, e.g. 2-dimensional for latitude and longitude) and t the time). Or trajectories can be recorded which follow e.g. the route of the weather balloon. These can then be represented as sequences or two-dimensional matrices.

3.4 Raster Data

Raster data is similar to point reference data. The difference is that the locations of the measuring points are fixed. Although the word "raster" might suggest otherwise, these locations do not necessarily have to form a grid. Also, the measuring interval does not need to be of a fixed length. An example for raster data would be the measurement of traffic using sensors at fixed locations. An exemplary record of such a measurement can be seen in Fig. 5.

Since raster data is discrete, it can be stored as time series information, spatial maps or a raster. Time series are stored for each location and can be represented by a sequence. Spatial maps are a collection of all observations at all locations for each discrete timestamp. They can be represented as graphs or two-dimensional matrices. A spatio-temporal raster contains a collection of spatial maps which chronologically follow each other. It can be represented as a two-dimensional matrix or a three-dimensional tensor. A matrix for example would be of size n by m with n being the amount of timestamps and m being the amount of measuring points. Every value v_{ij} in the matrix is then the measurement at the ith time and jth location.

3.5 Videos

Since the pixels of a video can be seen as the locations and the frames as the time, videos can also be considered

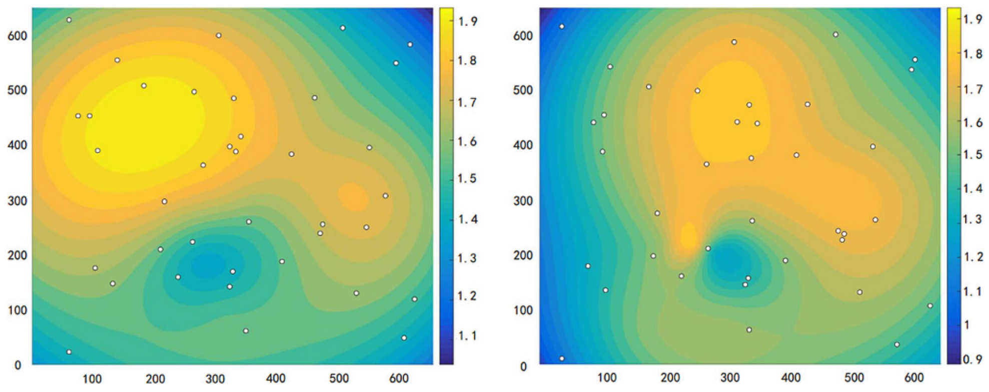

Fig. 4 Example for a set of point reference data [23]. In this example for point reference data, two timestamps are shown. For each timestamp, there are only values for the locations of the small white points. Based on them, the entire field can be approximated

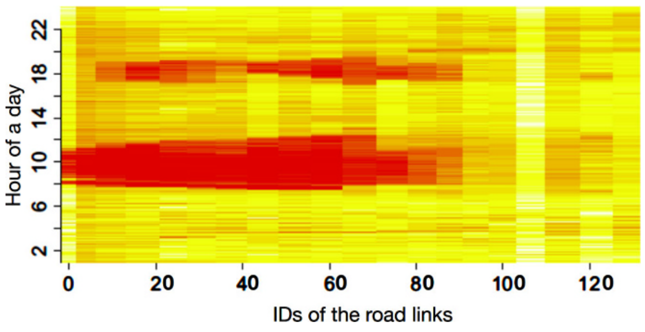

Fig. 5 Example for a set of raster data [23]. In this example for raster data, the traffic density of different road intersections is depicted. In this example, the location is pictured on the x-axis only, by the intersection's ID. This means that a lot of spatial information is only retrievable with a map showing in which relation these IDs are. The y-axis shows the temporal features of the data

as spatio-temporal data. They could also be seen as a very structured kind of raster data.

Since there is only a certain number of pixels with a certain density, the data is discrete. It can be stored as spatial maps (represented as graphs or two-dimensional matrices) or a spatio-temporal raster (represented as a two-dimensional matrix or a three-dimensional tensor).

4 Algorithms for Spatio-Temporal Data Engineering

Having a spatio-temporal dataset with features described in 3, the next step is then *Data Engineering* – transformation of the data into a form that a machine learning algorithm can work with. The representations listed in Fig. 1 can be a base for this transformation.

The further data preparation depends on the kind of learning algorithm that the data is to be used for. There are several algorithms which are commonly used for spatio-temporal data. Most of them are some sort of neural networks, such as Convolutional Neural Network (CNN), Graph Convolutional Network (GraphCNN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM) or Sequence to Sequence Transformation (Seq2Seq).

The *Feature Engineering* step is usually a part of Data Engineering. All the methods for feature selection we described in 2 can be applied to spatio-temporal data. The particular method depends on the selected machine learning algorithm and the desired complexity of the resulting model.

4.1 CNN

CNNs are usually used for image classification or similar tasks. They take a multidimensional cube of values as input. With images, this can be the two dimensions of pixels and a third dimension containing the red, green and blue values of the corresponding pixel's color.

This structure can be adopted also to spatio-temporal data. For example, spatial maps and rasters can be used as input. Both of them are a sort of two-dimensional map which can be considered as an image. If they are equally distributed, it works even better, no further preprocessing needs to be performed. Otherwise, they need to be transformed into a more homogenous form, e.g., by adding empty spaces in between the existing spaces. This task can sometimes increase the resulting model significantly. In this case, it might be easier to use GraphCNNs instead of CNNs.

On spatio-temporal data, CNNs focus on the spatial part, however, the temporal part can be included in different ways. One would be to run the model for each time stamp and include the time on a higher level. Another would be to include the time as a new dimension in the input data. Then, e.g. one spatial map for each timestamp can be put together as one input record.

4.2 GraphCNN

Basically, GraphCNNs work the same way CNNs do. However, they accept graphs as input. These do not need to have the same topology. A GraphCNN dynamically applies the operations to the graph's nodes which lie next to each other. As with CNNs, after one or multiple GraphCNN layers, a fully connected final layer can follow. This enables classification for specific nodes or regression, etc. Or, as an alternative, the embedded graph nodes can be considered themselves and used to classify the entire graph.

As with CNNs, the spatial part of the data is focused here. The temporal part can either be added by something on a higher level, or it can be integrated into the data. For example, a graph could contain different data for different times, included in the information each node saves.

It is self-explanatory that this kind of machine learning method is suitable for data represented by graphs. A good example is a traffic-flow.

4.3 RNN

All so far mentioned methods are mainly based on the spatial part of spatio-temporal data. RNNs and also LSTMs consider the temporal part more closely. They are best suited to sequence data such as time series.

Both, RNNs and LSTMs, can receive the spatial information for each timestamp, one after another. This way, they learn how the state changes over time and can predict it's state in the future. The basic difference between the two methods is that RNNs consider the recent past while LSTMs also consider the older past. [4] is an example for an RNN approach and [6] for an LSTM approach.

4.4 Seq2Seq

A Seq2Seq model is a way to translate a series into a series of another kind. It is widely used for language translation but it can also be used for spatio-temporal data.

The idea is that the dependencies in the input are captured. In a time series, these relations are sometimes significant. This is for example the case with the temperature. The temperature at a certain time and place is usually similar to the temperature at the same place a few minutes earlier or later. [15] for example uses Seq2Seq.

4.5 Further methods

The methods described above are some of the most common ones. However, they are not the only ones used. For example, simpler methods such as Restricted Boltzmann Machines (RBM) are efficiently used for specific tasks, as in [13]. Or another preprocessing step could be also to autoencode the data, as in [11]. And if one of these methods is not sufficient, multiple ones can be combined.

5 Spatio-Temporal Data Mining

All these mentioned machine learning algorithms have their specific use and goals. Which one should be selected depends on what is to be accomplished. The most relevant and frequent tasks are related to classification or prediction.

5.1 Prediction

Predicting in the context of spatio-temporal data usually means that a prediction for a certain point in time is made. This is the most common task. Thus, there are many approaches on this task existing. They contain several of the different machine learning methods in certain ways.

As an example, in the field of weather forecasting, predictions are based on the history of the weather in multiple areas. Since the temporal aspect is to be predicted here based on time series, RNNs and LSTMs are a popular choice. They can receive the weather records of the past days in order, and then produce the forecast for the upcoming days. This is shown e.g. in [25]. There, LSTMs are combined with Seq2Seq.

Another good example is traffic. In this case, the aim is to predict, when and where accidents might occur. To achieve this, there are many different approaches. One of them is ConvLSTM, which is a combination of CNNs and LSTMs. This approach is shown in [24].

5.2 Representation Learning

Since spatio-temporal data is often only a part of a bigger pool of features, it sometimes needs to be represented in a simpler or just in a different way. In this case, representation learning is used.

Mostly, the data is needed in a form which can be used as simple features in a next step. Therefore, representation learning extracts the essence of the input data and delivers it in the desired form.

Basically, autoencoders are an example for an algorithm achieving this goal. However, autoencoders usually already expect a fixed amount of numerical features.

Seq2Seq also is another tool for representation learning and is quite useful with time series. For example, a route (trajectory) of an object could be represented as a time series and then, using Seq2Seq, it could be mapped into a form which allows easier comparison. Such a process has been carried out in [14].

Other algorithms can also be used. For example, RNNs or LSTMs can keep track on the total state of a dataset over the time, making it one record instead of one per recorded timestamp.

5.3 Classification

Another relevant task is classification which is about choosing into which category a set of data belongs. A well known example for a classification task with spatio-temporal data is the recognition of diseases indicated by certain brain activities. For this task, e.g., fMRI recordings are used. They can be classified by a wide range of machine learning methods, including LSTMs, CNNs or GraphCNNs.

For example, an fMRI scan can be modelled as a three-dimensional raster. This can then be inserted into a CNN which predicts whether the scan shows a specific illness or not. This is explored in detail in [19].

5.4 Estimation and Inference

Sometimes, an estimation is needed, such as an arrival time for a trip or a temperature at a place without a thermometer. The first mentioned problem is an estimation and the second one is an inference.

Estimations can for example be calculated from spatial maps or trajectories. If we want to estimate an arrival time of a trip, we can use multiple ways. If the trip is represented as a spatial map, CNNs can be used. This is due to the fact that CNNs can use a multidimensional cube of values as input and a spatial map can be represented as one, as a matrix to be concrete. If it is a trajectory, RNNs or LSTMs are more appropriate, as they may run through every station step by step, summing up the travel time. An example for a recursive neural network estimating an arrival time is [26].

Inferences can be calculated in many situations. As mentioned before, a measurement can be estimated for a place that no actual measured value exists for. This can also happen using different kinds of neural networks, such as CNNs and RNNs. [5] shows an example for an air pollution inference which uses a system including an RNN.

5.5 Outlier Detection

As in all measured data, there are outliers in spatio-temporal data. Most of the time it is crucially important to find and analyse or extract the outliers. As with all other tasks, the way this is done highly depends on the data's domain.

Often, there are outliers in events, such as in traffic. Some traffic jams occur regularly while others only occur once. This may be due to an accident or a special event. In order to find these outlier events, CNNs as well as LSTMs are a common choice. [20] for example uses CNNs to accomplish this task.

In spatial maps for the weather data, e.g., there might be extreme values, such as super high temperatures or fast winds. These can also be analysed by methods such as CNNs as shown in [17].

6 Conclusion

Feature Selection is driven by several reasons. First, the most important reason is to make the model more accurate by removing unnecessary distractions. In addition, the whole process has the advantage that fewer features require less computing power for training and especially for prediction. This in turn increases the speed. And eventually, the models are easier to interpret, because fewer variables make it easier to understand what the individual variable is responsible for.

We presented in this paper a summary of feature selection methods. Although the methods can be used broadly, our focus has been more on spatio-temporal data. Therefore, other sections of the paper are devoted precisely to the topics of how to record the different kinds of spatio-temporal data, what algorithms can be used to process them, and for which types of analysis they are particularly well suited.

The presented work can serve as a roadmap for scientific projects and projects with industry, which deal with spatio-temporal data. Furthermore, we plan to systematically evaluate practical examples from our own projects and record further recommendation possibilities for the spatio-temporal data engineering process.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Akoglu H (2018) User's guide to correlation coefficients. Turk J Emerg Med 18(3):91–93
2. Atluri G, Karpatne A, Kumar V (2018) Spatio-temporal data mining: a survey of problems and methods. ACM Comput Surv. https://doi.org/10.1145/3161602

3. Chandrashekar G, Sahin F (2014) A survey on feature selection methods. Comput Electr Eng 40(1):16–28 (40th-year commemorative issue)
4. Cheng L, Zang H, Ding T, Sun R, Wang M, Wei Z, Sun G (2018) Ensemble recurrent neural network based probabilistic wind speed forecasting approach. Energies 11(8):1958
5. Cheng W, Shen Y, Zhu Y, Huang L (2018) A neural attention model for urban air quality inference: learning the weights of monitoring stations. In: The Thirty-second AAAI Conference on Artificial Intelligence, pp 2151–2158
6. Cui Z, Ke R, Pu Z, Wang Y (2018) Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction. arXiv preprint arXiv:1801.02143
7. El Aboudi N, Benhlima L (2016) Review on wrapper feature selection approaches. In: 2016 International Conference on Engineering MIS (ICEMIS), pp 1–5
8. Feizi-Derakhshi M-R, Ghaemi M (2014) Classifying different feature selection algorithms based on the search strategies. In: International conference on machine learning, electrical and mechanical engineering, pp 17–21
9. Genuer R, Poggi J-M, Tuleau-Malot C (2010) Variable selection using random forests. Pattern Recognit Lett 31(14):2225–2236
10. Ghavami P (2019) Big data analytics methods. De Gruyter, Boston
11. Huang H, Hu X, Zhao Y, Makkie M, Dong Q, Zhao S, Guo L, Liu T (2018) Modeling task fmri data via deep convolutional autoencoder. IEEE Trans Med Imaging 37(7):1551–1561
12. Jović A, Brkić K, Bogunović N (2015) A review of feature selection methods with applications. In: 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp 1200–1205
13. Larochelle H, Bengio Y (2008) Classification using discriminative restricted boltzmann machines. In: Proceedings of the 25th International Conference on Machine Learning ICML '08. Association for Computing Machinery, New York, NY, USA, pp 536–543
14. Li X, Zhao K, Cong G, Jensen CS, Wei W (2018) Deep representation learning for trajectory similarity computation. In: 2018 IEEE 34th International Conference on Data Engineering (ICDE), pp 617–628
15. Liao B, Zhang J, Cai M, Tang S, Gao Y, Wu C, Yang S, Zhu W, Guo Y, Wu F (2018) Dest-resnet: a deep spatiotemporal residual network for hotspot traffic speed prediction. In: Proceedings of the 26th ACM international conference on Multimedia, pp 1883–1891
16. Novaković J (2016) Toward optimal feature selection using ranking methods and classification algorithms. Yugosl J Oper Res 21(1):119–135
17. Racah E, Beckham C, Maharaj T, Ebrahimi Kahou S, Pal C et al (2016) Extremeweather: a large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events. arXiv preprint arXiv:1612.02095
18. Sánchez-Maroño N, Alonso-Betanzos A, Tombilla-Sanromán M (2007) Filter methods for feature selection: a comparative study. In: Proceedings of the 8th International Conference on Intelligent Data Engineering and Automated Learning IDEAL'07. Springer, Berlin, Heidelberg, pp 178–187
19. Sarraf S, Tofighi G (2016) Deep learning-based pipeline to recognize alzheimer's disease using fmri data. In: 2016 Future Technologies Conference (FTC), pp 816–820
20. Sun F, Dubey A, White J (2018) Dxnat – deep neural networks for explaining non-recurring traffic congestion. arXiv:1802.00002
21. Tallarida RJ, Murray RB (1987) Chi-square test. In: Manual of pharmacologic calculations. Springer, New York, pp 140–142
22. Vergara JR, Estévez PA (2014) A review of feature selection methods based on mutual information. Neural Comput Appl 24(1):175–186
23. Wang S, Jiannong C, Yu PS (2019) Deep learning for spatio-temporal data mining: a survey. CoRR, abs/1906.04928
24. Yuan Z, Zhou X, Yang T (2018) Hetero-ConvLSTM: a deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. Association for Computing Machinery, New York, NY, USA, pp 984–992
25. Zaytar A, El Amrani C (2016) Sequence to sequence weather forecasting with long short-term memory recurrent neural networks. Int J Comput Appl 143:7–11
26. Zhang H, Hao W, Sun W, Baihua Z (2018) Deeptravel: a neural network based travel time estimation model with auxiliary supervision, pp 3655–3661
27. Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. J Royal Stat Soc Ser B 67(2):301–320