

Euro Working Group on Transportation Annual Meeting 2025 - EWGT2025

Anomaly detection as modularity-based community detection

Jakub Sliachan^a, Katja Kircher^b

^a*VTI, Malvinas väg 6, 10215 Stockholm, Sweden*

^b*VTI, Olaus Magnus väg 35, 58195 Linköping, Sweden*

Abstract

When measuring how drivers overtake cyclists, one of the underlying problems is extracting the overtaking event from a time series of lateral distance readings. This note aims to describe a simple approach that seems effective in applications like ours. It consists of carefully transforming our problem into a network problem, then leveraging a community detection algorithm to extract subsequence candidates. Lastly, we choose the anomalous subsequence from the set of returned subsequences. To the best of our knowledge, this approach to anomaly detection does not appear in the literature even though it is intuitive, offers a fair amount of control, and is not computationally expensive. Our goal is to present the crux of the method with clarity and identify where more effort could improve it. We demonstrate our approach with modularity-based community detection and point out a shared nature of our approach with density-based cluster detection methods.

© 2026 The Authors. Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>)

Peer-review under responsibility of the scientific committee of the Euro Working Group on Transportation Annual Meeting 2025 - EWGT2025.

Keywords: anomaly detection; modularity; networks; overtaking cyclists; time series;

1. Introduction

Drivers overtaking cyclists on rural roads is a canonical example of a driver-cyclist interaction. It is an inevitable friction point in transport because, among other things, separate infrastructure in rural areas would be resource intensive to build and maintain. Hence, rural roads will likely remain shared by drivers, cyclists, and others. While numerous variables influence how an overtake is perceived by a cyclist, the lateral passing distance is key and we focus on it here. For background information on the topic of overtaking, see [Kircher and Niska \(2024\)](#), [Beck et al. \(2019\)](#), [Walker \(2007\)](#), [Nolan et al. \(2021\)](#) and references therein.

Several countries regulate how cyclists should be overtaken, typically by mandating minimum passing distances. These can range from 3*ft* in parts of the United States to 2*m* outside built-up areas in Germany and Austria, see [Kircher](#)

* Jakub Sliachan. Tel.: +46-72 229 6923

E-mail address: jakub.sliachan@vti.se

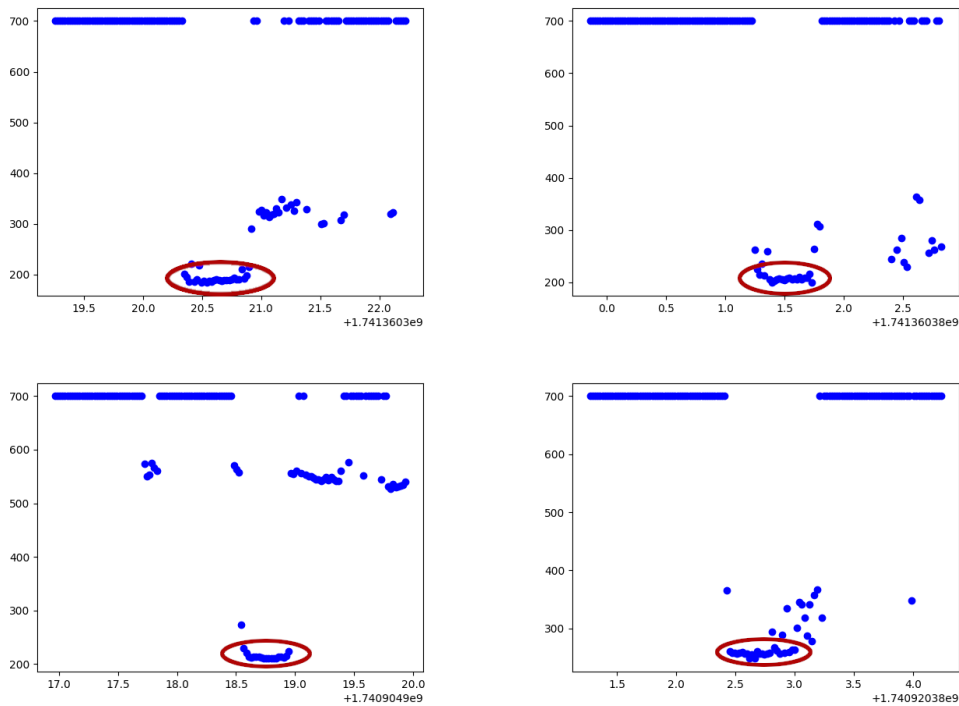


Fig. 1. Examples of lateral distance time series with overtaking events. Each is an excerpt ~ 3 seconds long. On the x -axis is time (unix timestamp), on the y -axis is lateral distance (in cm). The readings intentionally “max out” at $700cm$. The approximate part that we want to isolate in each case is circled.

and Niska (2024) for an overview. To assess compliance with the law and to address research questions like cyclists’ perception of the overtake depending on the passing distance, it is key to measure lateral passing distance reliably. The most typical way to do this is with a measuring device mounted on the bike.

We use a lidar device attached to the bicycle seat post. The lidar is pointing orthogonal from the bicycle towards the centre of the road so that the overtaking vehicles reflect the rays. The lidar takes lateral measurements indiscriminately at a fixed frequency, $50Hz$ in our case. This means that most of the lateral distance readings do not correspond to the distance between the cyclist and an overtaking vehicle, but rather to the distance between the cyclist and random objects on that side of the cyclist when no vehicle is overtaking. Therefore, the problem arises how to extract the overtaking readings from the sequence of all readings.

Lateral distance readings that we care about (when the vehicle is next to the cyclist) tend to be smaller than default lidar readings, which makes them anomalous in this context. Therefore, we approach the problem as anomaly detection. In general, detecting outlying observations is key in many applications. Sometimes one may want to remove the unrepresentative observations such as defective readings or extreme values caused by extraordinary events. Other times, like in our case, we are actually seeking anomalies as evidence of non-standard events such as overtakes – something that we want to observe. The situation can additionally become more complicated if some of the extreme values are of interest to us (valuable data) and some are not (noise). In such cases, it often comes down to exploiting contextual properties of the data.

Specifically, we are looking for a noisy outlier “cluster” in a noisy time series. Consider the examples in Figure 1. They all contain various types of noise, some stemming from road furniture, other from the dust cloud behind the overtaking vehicle, and sometimes bad readings come from see-through surfaces (windows) that do not reflect the lidar signal. A particular kind of “noise” is the presence of an oncoming vehicle around the same time as the overtake happens. In such a case we see an anomaly analogous to an overtake except with fewer readings (higher relative passing speed) and greater lateral distances (opposite side of the road), see Figure 2.

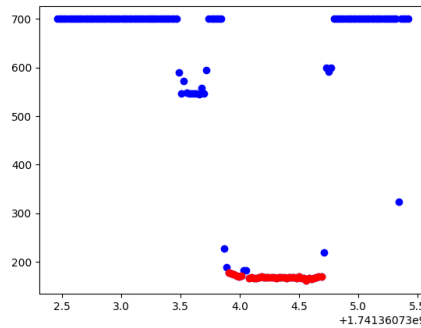


Fig. 2. Detected overtake (in red) with an oncoming vehicle having passed the cyclist just before that.

This paper presents a simple method to extract the lateral distance readings corresponding to the passing vehicle from the set of all readings. Due to large amounts of data it is necessary to do this in an automated (or semi-automated) way such that false positives and false negatives are infrequent.

2. The method

The main contribution of this note is to illustrate how intentional choice of distance function and consequent rephrasing of the problem in terms of weighted graphs (networks) allows one to use network-based community detection methods even for data that is not graph-like. Our approach can be summarized in four steps.

1. *Reduce* the neighbourhood of the event as much as possible.
2. *Construct* the corresponding graph with carefully chosen edge weights.
3. *Partition* the graph into communities and choose the right community.
4. *Clean* the chosen community further as needed.

We are using modularity as the score for determining how clustered a network is. One could experiment with other measures of “clusteredness” as it would be interesting to see how robust (or not) our approach is to the choice of scoring measure. However, that falls outside of the scope of this note.

We now define modularity, a measure of the density of edges within communities with respect to the density of edges between the communities. Given a vertex partition \mathcal{P} of a graph G with vertex set V and edges in E , consider $P \subseteq V$, and let $e(P)$ denote the number of edges in a subgraph induced (in G) by P , i.e. with both endpoints in P . Furthermore, let $ds(P)$ denote the sum of degrees in G of all vertices in P . Then the modularity $q_{\mathcal{P}}(G)$ of a partition \mathcal{P} of G is defined (see [Newman and Girvan \(2004\)](#)) as an “edge contribution” minus the “degree tax”, the two competing terms of (1).

$$q_{\mathcal{P}}(G) = \frac{1}{m} \sum_{P \in \mathcal{P}} e(P) - \frac{1}{4m^2} \sum_{P \in \mathcal{P}} ds(P)^2. \quad (1)$$

Modularity of a graph G , on the other hand, is the maximum $q_{\mathcal{P}}$ over all partitions \mathcal{P} of G

$$q_{\mathcal{P}}^*(G) = \max_{\mathcal{P}} q_{\mathcal{P}}(G). \quad (2)$$

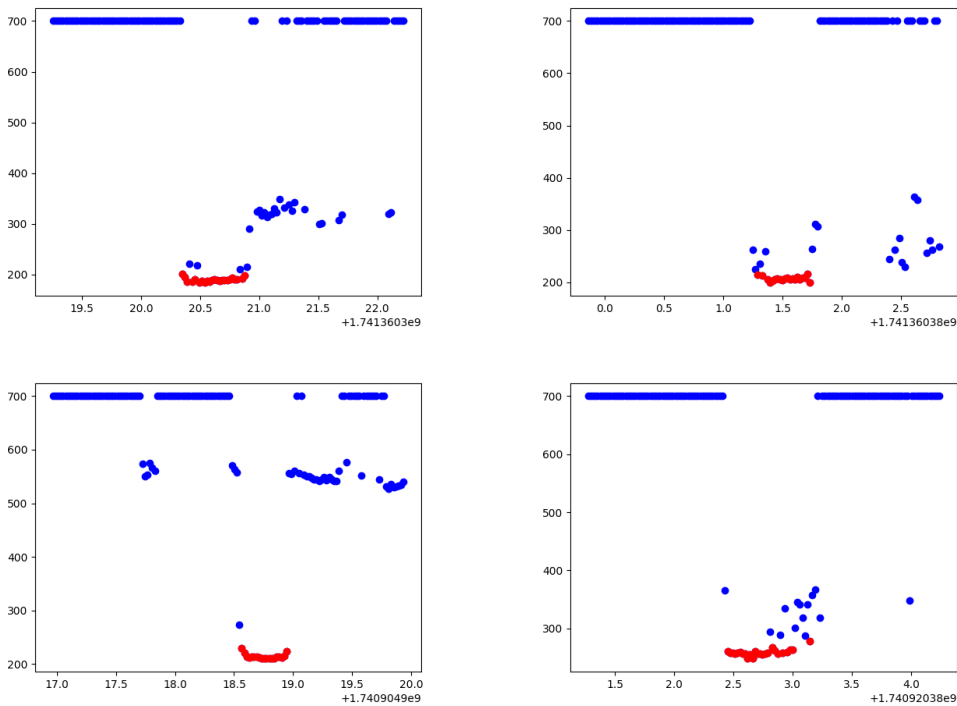


Fig. 3. Examples of data with detected “overtakes” by a crude version of our method, but we kept only the values close to the median in the selected community.

In the Louvain algorithm, modularity score informs the grouping of vertices into parts by measuring how clustered the graph is under each considered partition, i.e. the goal is to maximize the modularity score.

2.1. Reduce the problem size

For complexity reasons, it is desirable to reduce the scope of the problem as much as possible before any computations occur. We managed to identify a neighbourhood of each overtake in the long time series spanning a bike ride of multiple hours. Each of these neighbourhoods is roughly 3s long at 50Hz, so between 150 and 200 readings.

Additionally, in our case we can leverage that the desired anomalies have below-average values in this excerpt of readings. This is a consequence of the design (defaulting readings to high values) and the nature of the problem (overtaking cars tend to be the closest objects to the cyclist on the road-side of the cyclist). This way of reducing the excerpt to below-average values significantly improves our approach. However, to demonstrate the method as clearly as possible, we will omit this step.

In practice, it seems to be the case that, for instance, Louvain modularity-based algorithm for network community detection runs in a time proportional to the number of edges, see [Blondel et al. \(2008\)](#). There are other algorithms, see [Lancichinetti and Fortunato \(2009\)](#) for an overview, but our situation is not affected by the resolution limit of modularity, our communities are non-overlapping, and badly connected communities are unlikely to arise given the character of our data and the distance function on it. So for our purposes, the Louvain method is sufficient and the reasonable runtime is a welcome benefit. For more information on the method, see [Lancichinetti and Fortunato \(2011\)](#). Constructing G itself means touching every pair of vertices in V and is therefore in $\Omega(n^2)$, so Louvain method is not an issue complexity-wise.

At this point, we need to mention the Leiden algorithm, see [Traag et al. \(2019\)](#), which is a strict improvement over Louvain in terms of runtime, connectedness of communities, and resolution limit issues around small communities. We are likely to deploy `leidenalg` instead of Louvain in future iterations of our detection methods.

2.2. Construct the graph

This is the crux of the method where, in essence, we rearrange the data points through a carefully picked distance function to then suit a sort-of density method. We do this in the language of graphs to be able to use modularity score to identify communities (clusters), but one could arguably use other density-based anomaly detection methods to identify clusters within these transformed data points. One could consider various modifications of k -nearest neighbours approach through local distance-based outlier detection methods like the one suggested by Zhang et al. (2009) or one of the “density-based spatial clustering of applications with noise” algorithms such as those in Ester et al. (1996).

Let $G = (V, E)$ be a simple undirected graph with a vertex set v and an edge set $e = \{uv \mid u, v \in V\}$. Given that we switch between graph and network language, we use equivalent terms interchangeably: vertex–node, edge–connection, graph–network, etc. Additionally, let $N(v)$ denote the set of neighbours of a vertex v , i.e. $N(v) = \{u \in V \mid uv \in E\}$.

Now, let v_1, \dots, v_n be the readings in the data excerpt containing an overtake, each $v_i = (x_i, y_i)$, ordered chronologically. We construct a weighted complete graph K_n with vertex set $V = \{v_1, \dots, v_n\}$ and edge set $E = \{v_i v_j \mid v_i, v_j \in V\}$ such that the weight of each edge corresponds to the distance of the vertices incident with it, $w(v_i v_j) = d(v_i, v_j)$ for some distance function d . An obvious choice is Euclidean distance (L_2 -norm) with a few tweaks. Alternatively, one could simply distinguish between edges ($w = 1$) and non-edges ($w = 0$). Depending on the connectedness of the resulting graph, modularity-based detection may be unnecessary and it might be possible to simply partition the graph into connected components. Either way, the general approach is to pick a good function d and partition the graph according to the weights. For illustration, let us consider both a simple Euclidean-based distance function and a 0–1 (edge/non-edge) function.

Recall that the vertices in our graph are canonically ordered given that they come from a time-series. This is important, because we will use the fact that if two vertices were already seen, then we know whether they are adjacent or not. Additionally, we assume that the x -axis and y -axis scale similarly and the values are of similar magnitudes. Otherwise we need to treat the disparity between impact of x and y coordinates on the distance/weight.

Let $v_i, v_k \in V$ be two vertices of K_n , such that $i < k$. Let d_e denote a distance which is inversely related to the Euclidean distance and slightly parametrized for additional control. Then let d_b denote the distance that is either 0 or 1, based on proximity conditions along x -axis and y -axis. In short, in d_b , the edge $v_i v_k$ is weighed 1 if v_i, v_k are close to each other along y -axis and either close also along x -axis or there is another point v_j such that $v_i v_j$ has weight 1 and v_k is close to v_j along x -axis. Precise statements are in (3) and (4).

$$d_e(v_i, v_k) = \frac{c}{\|v_i - v_k\|_2^\alpha} \quad (3)$$

$$d_b(v_i, v_k) = \begin{cases} 1 & \text{if } |y_i - y_k| < a \text{ and either } |x_i - x_k| < b \text{ or } \exists j < k : v_j \in N(v_i) \text{ and } |x_j - x_k| < \epsilon b \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Distance d_e is parametrized by $c > 0$ and $\alpha > 0$ while d_b is parametrized by $a, b, \epsilon > 0$. In practice, we take $c = 100$ and $\alpha = 3/2$. For d_b parameters, we let $a = 30, b = 0.3$ and $\epsilon = 0.6$.

The role of the distance function is to redistribute the points in the space such that the resulting spread is more reflective of the semantic relationships between the points, as perceived by the community detection method to be applied. For instance, if we decide that two points u and v are such that $d(u, v) = 0$, then they are indistinguishable from each other by any distance- or density-based method and consequently, will always be assigned to the same community. In other words, the distance function is key and has to be chosen with the detection method in mind.

Recall that d_e and d_b are merely examples of weight functions one can use given the character of the data, properties of the sensors, and the goal of the data analysis. As is, d_e is a “broad brush” weight function which leverages no specific properties of the data. Contrast it with d_b , which is an example of a data-specific weight function leveraging a few real-world parameters: a being the vertical spread of the anomaly, b being the “duration” of the overtake including an

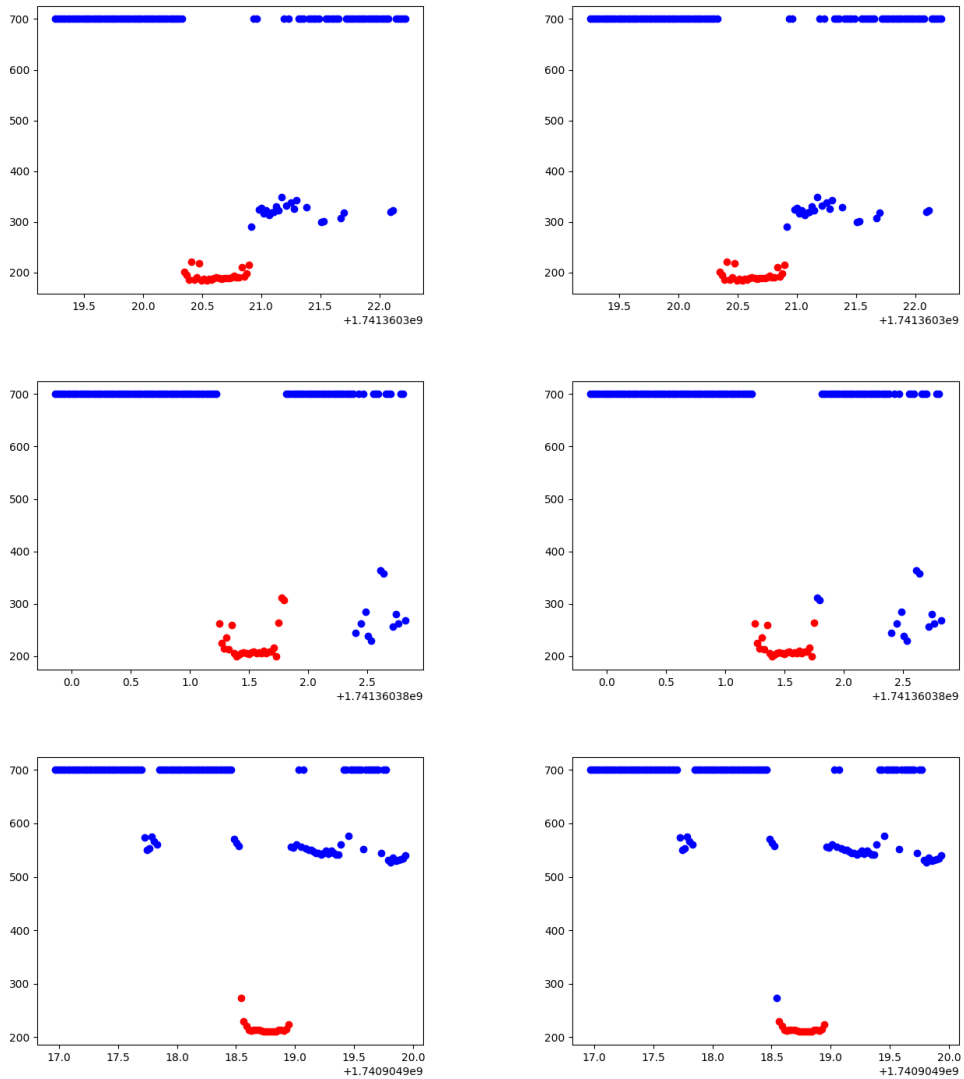


Fig. 4. Comparing continuous weights d_e (left) with binary weights d_b (right). All of them were identified correctly except for filtering out small local noise. This can be readily corrected with basic methods in post-processing once the sought “community” has been identified.

allowance for chaining through neighbours-of-neighbours controlled by the ϵ parameter. In short, there is significant freedom when choosing the weight function.

2.3. Partition the graph

This step simply runs Louvain algorithm on the graph obtained in the previous step. As mentioned earlier, there are other algorithms to deploy at this stage, most notably Leiden algorithm. We then pick the largest community not corresponding to the default (maxed out) values. Note that picking the right community from the partition of G could be a hard problem. However, the character of our data implies that always choosing the largest community from the sub-average readings in the excerpt is a reliable approach.

2.4. Clean the community in post-processing

In practice, there are occasions when even small deviations from ideal detections need to be avoided. In such cases, certain post-processing steps might need to be taken on the detected community. A possible example is taking the largest clique in the community and only reattaching to it other vertices from that community if and only if they meet certain strict criteria, stricter than are possible when dealing with coarse high-level communities. Or one could simply exclude values from the detected community that deviate sufficiently from the mean, median, etc. See Figure 3 where we show a simple improvement over high-level detections in Figure 4. The problem of eliminating outliers within the detected community is likely simpler than identifying the right community.

3. Limitations

The main limitation of this note is the absence of benchmarking of the proposed method. However, this is a smaller omission than it seems. Our method is a reduction of the original problem to a problem in a different area where methods are well established, benchmarked, and studied. In this regard, our method does not lack comparison to other existing tools as modularity-based community detection has been studied in the context of other community detection methods. The novel part of our method is the reduction, which we tried to illustrate depends heavily on the character of the data. To assess the “accuracy” of the detections we would need to obtain reliable ground truths. In many cases, for the lidar readings this is difficult even for humans. We would like to see a controlled study (in a lab) fixing precise timestamps for when an overtake begins and ends, then using these external time delimiters to cut out “ground truth” events from our lidar sequence and comparing them with detections obtained via our method. Unfortunately, this is well outside the scope of our project and is therefore left to the community.

Other issues are mostly centered around post-processing. After we are satisfied with our partition, there is still the problem of choosing the right community to be the desired outlier. This could be a difficult problem in itself, as the differences between communities could be subtle enough that telling apart good choices from bad choices is difficult. Then there is the problem of choosing the distance function. In our case, we should create equivalence classes of points which are in a contiguous sequence with every two consecutive points very close to each other. Otherwise a long sequence of consecutive points will be split into sub-communities due to large distances of the outer points. Also, depending on the specific application, different “clusteredness” scores and detection algorithms might need to be used, for instance, if sensitivity to individual points is important.

4. Conclusion

The method we present is quite particular, but does come with inbuilt flexibility. In our case, we leverage the fact that sequences of points can be converted to cliques in a straightforward way, while cliques form the communities in networks. The lack of sensitivity to single points is a strength for our use. The adaptability comes from parametrized design: distance function when constructing the graph, thresholds, and parameters of the community detection algorithm.

From a purely methods perspective, it would be interesting to compare our approach with established machine learning approaches such as those outlined in Schmidl et al. (2022), e.g. variations of the long short-term memory algorithm. Another direction to explore is to vary our approach through the use of different community detection methods (not Louvain or not modularity), including spectral approaches relying on the Laplacian matrix of the graph and its eigenvalue decomposition. See Newman (2013) and von Luxburg (2007). The k -nearest neighbours methods are also based on distances and could be used.

Acknowledgements

The authors would like to thank Jonathan Nolan for useful discussions about limitations of ultrasound sensors and for sharing raw data samples from Nolan et al. (2021) with us.

References

- Beck, B., Chong, D., Olivier, J., Perkins, M., Tsay, A., Rushford, A., Li, L., Cameron, P., Fry, R., Johnson, M., 2019. How much space do drivers provide when passing cyclists? Understanding the impact of motor vehicle and infrastructure characteristics on passing distance. *Accident Analysis & Prevention* 128, 253–260.
- Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E., 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008.
- Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise, in: *KDD*, pp. 226–231.
- Kircher, K., Niska, A., 2024. Overtaking cyclists in mixed traffic : Knowledge basis for recommendations for safer cycling. Technical Report 1189A. Swedish National Road and Transport Research Institute, The Human in the Transport system.
- Lancichinetti, A., Fortunato, S., 2009. Community detection algorithms: A comparative analysis. *Physical review. E, Statistical, nonlinear, and soft matter physics* 80, 056117.
- Lancichinetti, A., Fortunato, S., 2011. Limits of modularity maximization in community detection. *Physical Review. E* 84, 066122.
- von Luxburg, U., 2007. A tutorial on spectral clustering. *Statistics and computing* 17, 395–416.
- Newman, M.E.J., 2013. Spectral methods for community detection and graph partitioning. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics* 88, 042822.
- Newman, M.E.J., Girvan, M., 2004. Finding and evaluating community structure in networks. *Physical Review. E* 69, 026113.
- Nolan, J., Sinclair, J., Savage, J., 2021. Are bicycle lanes effective? The relationship between passing distance and road characteristics. *Accident Analysis & Prevention* 159, 106184.
- Schmidl, S., Wenig, P., Papenbrock, T., 2022. Anomaly detection in time series: A comprehensive evaluation. *Proceedings of the VLDB Endowment (PVLDB)* 15, 1779–1797.
- Traag, V., Waltman, L., van Eck, N., 2019. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports* 9.
- Walker, I., 2007. Drivers overtaking bicyclists: Objective data on the effects of riding position, helmet use, vehicle type and apparent gender. *Accident Analysis & Prevention* 39, 417–425.
- Zhang, K., Hutter, M., Jin, H., 2009. A new local distance-based outlier detection approach for scattered real-world data, in: *Advances in Knowledge Discovery and Data Mining: 13th Pacific-Asia Conference, PAKDD 2009 Bangkok, Thailand, April 27-30, 2009 Proceedings* 13, Springer. pp. 813–822.