GIMA

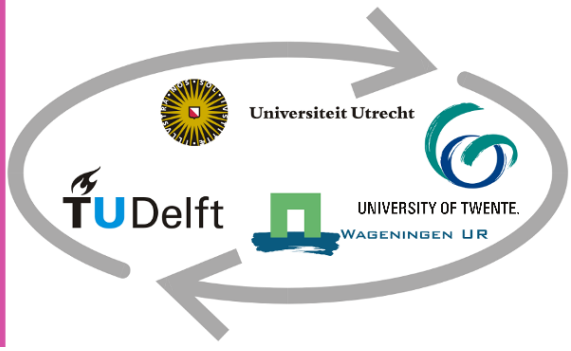
Geographical Information Management and Applications

Master thesis

Investigation of attack strategies on geoprivacy with spatial obfuscation

Maarten Mol

27-02-2019



Abstract

This research looks into the current state of spatial obfuscation algorithms and investigates attack strategies to circumvent them. A set of over 37.000 cyclist tracks are obfuscated using a variety of obfuscation algorithms, after which two categories of attack strategies were applied to reconstruct the original tracks based on the obfuscated tracks. These attack strategies were based on heuristic approaches on the one hand, with a deep learning approach to privacy attacks on the other. Using an evaluation measure determining the overlap between actual tracks and predicted tracks, each attack strategy was evaluated, revealing the applied deep learning approach not to be suitable as an attack strategy in its current form, and showing heuristic methods to function better. However, these methods are still unable to recover the original track completely, and further research is required to get attack strategies suitable to evaluate the performance of spatial obfuscation algorithms.

1 Introduction

With a growing number of consumers in possession of a smartphone, location-based services like Strava and Google maps have started to attract more customers. These types of location-based services (LBS) provide a lot of functionality to their users based on their geo-location. In recent years, however privacy has become more of an issue. Strava, for example, raised controversy with their published heat-maps revealing the location of secret U.S. army bases (Hern, 2018).

However, it is not just secret army bases that should care about location privacy. Few people realise just how much information can be gleaned from their location data or the risks that could be involved if it fell into the wrong hands (Krumm, 2009; Kaasinen, 2003). Nevertheless, a person's location data can pose a significant threat, possibly exposing a user to unsolicited advertisements, scams, or even being tracked down with intent for physical violence (Shokri et al., 2011; Wordsworth, 2015). Furthermore, many mobile phone applications nowadays actually gather location data, regardless of whether it is relevant to the application itself (Thurm and Kane, 2010).

Location privacy, when compromised, can have a significant impact on the users of an LBS. Naturally, the easiest way to protect location data is to not gather or store it in the first place. However, an LBS heavily relies on location data to function and location data can provide significant value to scientific research. A study by Ghinita (2009) identified these two types of scenario where location privacy plays a large role. The first is that of the online LBS, where users are provided with a live service based on their location. There is, however, also a second relevant domain for location privacy, namely that of research, where gathered location data can be published without endangering the privacy of the individuals that data was gathered from.

The question then becomes how best to protect a user's location privacy. A part of this protection is generally done through network security, prevent-

ing outsiders from unlawfully obtaining data. This does not however entirely exclude the possibility of a company's data being hacked or obtained through social hacking. This is the process of extracting information from individuals through manipulation or deceit (Nolan and Levesque, 2005).

Similarly, network security does not protect against insiders such as employees that may want to exploit the data. Finally, companies have also started to publish their own data through open data protocols such as the European INSPIRE framework. This could be a big step towards an open data ecosystem which does not compromise privacy issues for commercial or other uses, if it was possible to publish location data without compromising user privacy, even at a less precise resolution than the original.

Currently, one of the most commonly used location privacy protection measures (LPPMs) is to aggregate data to a lower resolution. While this generally works well for privacy protection, it also distorts the data so that patterns in the original data are no longer present for investigation (Zandbergen, 2014). Therefore, a lot of research has gone into creating different methods for privacy protection which maintain as many of the original data characteristics as possible. The methods used to protect a user's privacy are generally called obfuscation algorithms. This research will look into those obfuscation measures that distort the data sufficiently to protect the original data, while still keeping it mostly intact.

1.1 Problem Definition

There are many different ways in which data may be obfuscated. Location data can be moved, rotated and distorted in any conceivable number of ways. The value, however lies in how well-suited an obfuscation method is for a specific purpose. For example, if one were to find a theoretical obfuscation algorithm that perfectly protects the privacy for a data-set but distorts the data beyond any utility, it is not a useful obfuscation method. Similarly, if we have an incredibly low distortion, no matter how complex our algorithm, it is still virtually useless if it does not protect adequately against possible attackers.

This highlights a problem with current research into obfuscation algorithms. While most research is focused on creating and improving obfuscation algorithms with acceptable distortion, possible attack strategies are generally only considered implicitly (Fechner and Kray, 2012). This leaves a significant gap in the current body of scientific knowledge, where location protection is discussed without considering how an attacker might go about computationally circumventing these obfuscation methods. In that sense, obfuscation algorithms have so far been developed in a sort of academic vacuum, where it is attempted to protect the privacy from users without knowing what it needs to be protected from. This research aims to take the first steps in exploring what types of location-based attack strategies are possible for user trajectories, to gain a better insight into how to defend user privacy in an LBS.

As a consequence of the limited research into attack strategies, the evaluation of different obfuscation algorithms is also limited. While there is some research

into the quantification of obfuscation algorithm security (Shokri et al., 2011), it is still mainly theory-crafting, rather than application-based. This research will therefore look into an application-based approach of evaluating obfuscation strategies by investigating possible attack strategies and examining how these spatial obfuscation algorithms hold up. In order to study the performance of different obfuscation algorithms, as well as attack strategies, developing a performance measure will also be a part of this research.

1.2 Context

This research will be conducted in the context of the Global Geo-Health Data Center (GGHDC, `https://globalgeohealthdatacenter.com`) setting up an LBS where users can upload point data (often tracks) to receive high-quality information concerning the air quality at that location and time. Air quality is shown to have a strong impact on health (Neidell, 2004), and furthermore, air quality is highly localised (Padilla et al., 2014). Therefore the balance between user privacy and location precision which plays out in obfuscation algorithms (Seidl, 2014) is highly relevant to this field.

In order to protect the privacy of its users, the GGHDC wants to obfuscate spatial data before sending it to the service while still maintaining high data quality and low distortion, to provide as much value as possible to users.

The context of this research will assume the product with an attacker that has a set of obfuscated data from which he or she wants to recover the original points. To achieve this, the attacker is assumed to possess a set of unobfuscated tracks that can be used to reverse engineer the applied obfuscation algorithm. This can be done by sending the unobfuscated data to the server for testing and receiving the obfuscated enriched data, or by using an obfuscation script that has been made available by the GGHDC for privacy protection before uploading. Aside from the likelihood of such a scenario, it was also chosen to allow for more targeted attack strategies to be applied, before starting research on a blind obfuscation problem.

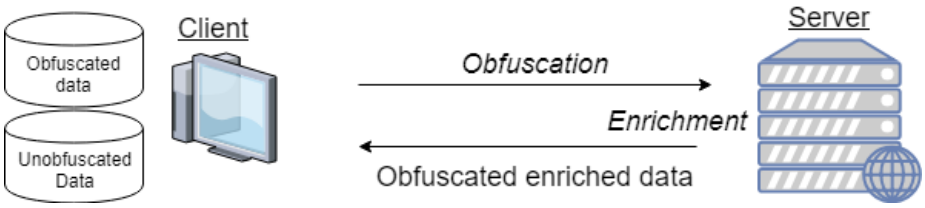


Figure 1: The context of the attack strategies in this research

2 Research Objectives

This research aims to collect, formulate and test attack strategies against spatial obfuscation methods. This research into attack strategies will improve the testing possibilities for new obfuscation methods, as well as allow obfuscation measures to be compared based on their performance against such attack strategies.

This research will formulate attack strategies to a collection of different spatial obfuscation algorithms, examine ways of evaluating the obfuscation algorithms' performance, and apply this to a range of obfuscation algorithms. This is done with the goal in mind of creating a testing toolkit for spatial obfuscation algorithms through a suite of attack strategies that can be applied to test a new spatial obfuscation algorithm.

2.1 Research questions

The previously described research goal can be translated into the following research question:

> What types of attack strategies can be used against spatial obfuscation algorithms, and how can their performance be evaluated?

This can subsequently be split up into the following subquestions:

- What attack strategies can be used to counteract spatial obfuscation algorithms?
- How can attack strategies be evaluated for their performance on different obfuscation algorithms?
- Which attack strategies work best on what obfuscation methods?

2.2 Scope

In order to keep the research both relevant and feasible, some scope limitations have been applied.

Firstly, although many privacy protection strategies are based on aggregation, these will not be taken into account in this research, since a study done by Gruteser and Hoh (2005) shows that even aggregated spatial information can be separated into the original users that provide it. Furthermore, the data distortion in aggregation is too high to qualify as a viable privacy protection measure for highly localised air quality data. Additionally, the data distortion that occurs in aggregation makes subsequent analysis of aggregated data less valuable, since patterns such as clustering or localised spatial patterns are lost (Seidl et al., 2015).

3 Theoretical Framework

This section will contain the relevant information from literature in the field of spatial obfuscation and privacy methodologies, as well as providing the scientific background and concepts that are used in this research.

3.1 Spatial Obfuscation

This section will contain an overview of the relevant literature behind commonly used track obfuscation methods that have been selected for evaluation in this research.

3.1.1 Grid Masking

Grid masking is an obfuscation algorithm first introduced by Leitner and Curtis (2006). Grid masking creates a grid around the original data, after which points are manipulated within their respective grid cells. This maintains a large part of the spatial patterns available, while still providing a measure of privacy. The most common form of grid masking is to snap all points to their respective grid cell corners (Krumm, 2007; Seidl et al., 2015). This results in an approximation of which locations have been visited, but with an uncertainty equal to the applied cell-size.

3.1.2 Random Perturbation

In a random perturbation algorithm, points are offset from their origin by a random value Kwan et al. (2004). This random value adds noise to the data, which makes it more difficult to find the original route, while also keeping the data distortion relatively low. With a dense road network, such as in most populated areas in the Netherlands, it is easy for such distortions to mask which road the user has taken. An example of an obfuscated track is given in figure 2.

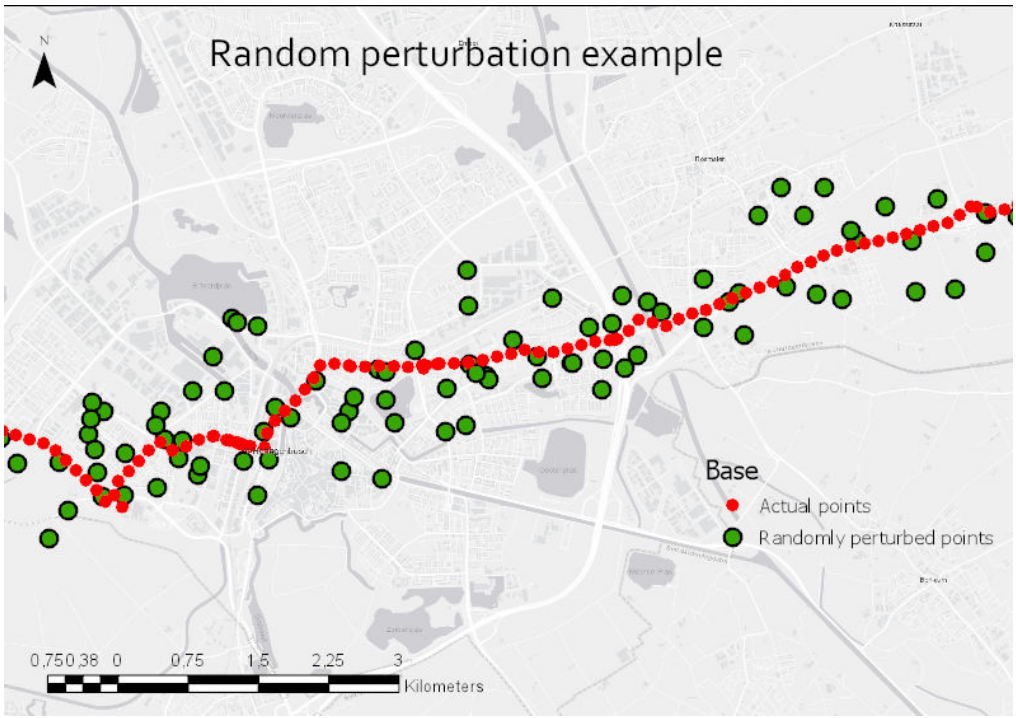


Figure 2: An example of random perturbation applied to a track

3.1.3 Crowding

Crowding, as first described in Kido et al. (2005) and detailed further by Scheider et al. (2019), is the concept of populating an actual track with a number of additional points to ensure the original track is no longer distinguishable from other possible tracks. The most important consideration is that added points should together form a credible track. Simply put, if a track is obfuscated with a crowding algorithm that leads into a body of water or straight through other physical obstacles, an attacker will easily be able to distinguish the original track from the fake one.

A different way to look at this crowding algorithm is that it simulates a form of k-anonymity. K-anonymity is the concept that any given data-point can be attributed to at least K other users, granting each person belonging to that data-point a certain degree of anonymity (Sweeney, 2002). In a crowding algorithm, this is no longer done by combining actual data-points, but noise is generated to artificially generate a k-anonymous track storage.

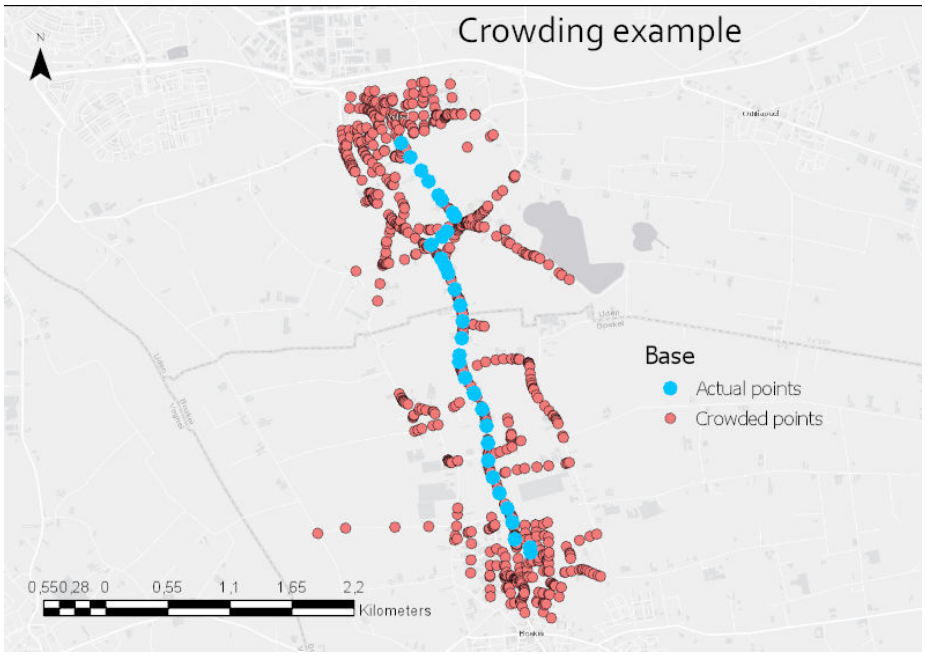


Figure 3: Crowding example applied to a track

In order to circumvent a crowding application, there are a few possible attack strategies. The first of which is to reverse-engineer the original obfuscation algorithm through the application of machine learning. By training a classifier to recognise which points have been added (by feeding in some initial training data), it should be possible to predict which points have been added as an obfuscation measure.

A different option is to check the credibility of the different possible tracks, generating a list of probabilities for each possible track, and selecting the track that is most likely. This selection can be made on the basis of track shape, movement credibility and location credibility.

3.2 Attack Strategies

The formulation of attack strategies will be done from the viewpoint of a malevolent insider with knowledge of the types of obfuscation methods available and access to the algorithm to run his own training set through it, to obtain paired combinations of actual and obfuscated tracks.

The attacker is assumed to have a set of obfuscated tracks he has obtained through other means, and his goal is to re-identify these to their original tracks as closely as possible. This approach is also called red teaming, and it is commonly used to identify weaknesses in security systems (Wood and Duggan, 2000). The

following sections detail the relevant theory concerning the attack strategies produced in this study.

This research will take two general approaches for attack strategies, where this section will contain the necessary literature background on which they are based. The first approach will be referred to as the heuristic approach, where methods are manually specified to narrow down the possibility space for the original track. The second approach will be based on machine learning, by applying neural networks to the data-set. This approach will be referred to as deep learning.

3.2.1 Heuristic methods

This section will contain the relevant literature behind the applied heuristic attack strategies.

Time series analysis

When you take away the background spatial context, a GPS track can also be viewed as if it were a time series (Hu et al., 2013). The main difference with time series, as opposed to GPS tracks, is that the goal is to predict both the X and Y values, as opposed to time series analysis predicting a Y value based on an independent X.

This approach has been employed in several studies to decrease the GPS error(Mao et al., 1999; Williams et al., 2004), but can theoretically also be applied as an attack strategy. In these studies, the main method used was a maximum likelihood estimation (MLE).

Another commonly used way of removing noise from time series data is the moving average (Hamilton, 1994). By assuming that change between observations is gradual, and affected by noise, a moving average can reduce the impact of noise on the observations, while still maintaining most of the gradual, actual change.

3.2.2 Deep learning

Although machine learning is a wide field with an equally wide range of possibilities, standard machine learning algorithms tend to have a limited capacity for learning complex interactions and nonlinear relations. Additionally, "simple" machine-learning requires extensive feature engineering to get the most descriptive features for the variable of interest (LeCun et al., 2015). By applying deep learning, further explained in the coming section, feature engineering is automated through back-propagation, and complex interactions can be modelled by specifying an appropriate network architecture.

A neural network can be seen as a series of connected nodes. When a layer in the network receives an input, the network multiplies the input with a set of weights, after which often a non-linear transformation is applied (Haykin and Network, 2004). The result of this is then passed on to the next layer, and so on until the network reaches the output layer, which is then used for evaluation.

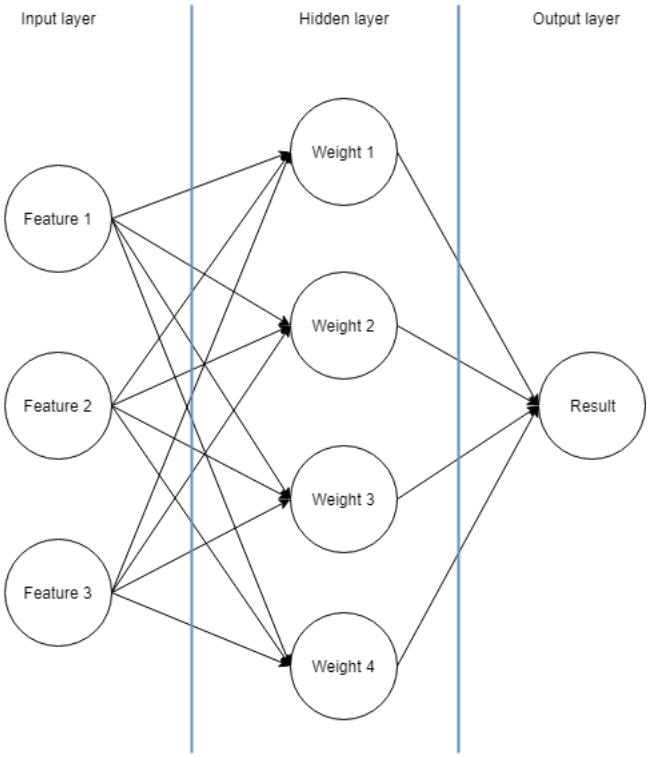


Figure 4: A basic illustration of a single-layer neural network

When the network is evaluated, the outputs are compared to the desired outputs in a process called backpropagation (Werbos, 1990). For this operation, the gradient of the previous operations with respect to the output loss is recursively evaluated, and the weights adjusted based on this and a specified learning rate parameter. In this way, the neural network will adapt to the type of operation needed to go from the training input to the training output and, if it is not overfitted to the training data, be able to predict new examples accurately.

Applying the theory to this research, we can input a track that has been obfuscated, and gather relevant features for each point in the obfuscated track. Using matrices, we can input the spatial distribution, and for a single point, we can use features such as distance from the nearest road or distance from the previous point.

These features are then fed into the model, and backpropagation is applied to adapt the weights in the model to reduce the training error, and thus improve results. This is done multiple times with the entire available data-set, each pass being called an epoch, until the specified number of epochs is reached or, when specified, the loss no longer decreases for a certain number of epochs.

Recurrent Neural Networks

Although neural networks are powerful, the previously described options are still the basics of neural networking. In addition to this, multiple types of more advanced layers have been developed. One such layer that is relevant to this research is the recurrent layer.

A recurrent layer, in its initial form, is very similar to any other layer commonly used in a neural network. It has a set of weights, an activation function, and its performance is improved through backpropagation. Its main application is for data that comes in some sort of sequence. Rather than applying its weight multiplication and activation function to the entire input at a time, the input is split up into its sequences, and passed through step by step (Grossberg, 2013). This allows the recurrent layer to build up a representation of the input as it receives it, allowing for memory units within the recurrent layer to retain some sort of general representation of the input sequence. Two commonly used recurrent layers are made up of long short-term memory units (LSTMs) (Gers et al., 1999) and gated recurrent units (GRUs)(Chung et al., 2014). The most important difference is that LSTMs are more complex in their calculations, sometimes allowing for better performance. On the other hand, GRUs are slightly less complex and thus train faster while still achieving nearly the same performance (Chung et al., 2014).

convolutional networks

A different type of layer used in this research is the convolutional layer. Although there are variations from 1 to 3 dimensions, only the 1-dimensional variant is relevant to this research, as the others are generally used for image or video processing (LeCun et al., 1995).

In the case of the 1-dimensional convolutional layer, the model has a vector of weights to optimise for, generally referred to as a filter. In a convolutional layer, the model overlays each filter over a part of the input sequence. The input values are then multiplied by the weight values in the filter, and summed to generate an output for the first input value. The filter is then moved up a step to the next input values each time, thus performing operations on the entire input sequence. After this is completed, a non-linearity is applied to the output values.

Precision and Recall

Two common evaluation metrics in machine learning for binary variables are precision and recall. Rather than just using accuracy (the difference between actual and predicted), it allows one to distinguish between the ratio true positives and false positives for precision, or true positives versus false negatives for recall (Powers, 2011). This can be captured in the following equations:

$$Precision = \frac{True\ positives}{True\ positives + False\ positives} \tag{1}$$

$$Recall = \frac{True\ positives}{True\ positives + False\ negatives} \tag{2}$$

Put more practically, table 1 contains some randomly generated results for a classification algorithm, where we compare the ground truth (denoted as actual) to the predicted scores. Calculating the precision is then done by calculating 8 / (8 + 14) = 0.364, and the recall comes out to 8 / (8 + 15) = 0.348.

	Predicted True	Predicted False
Actual True	8	15
Actual False	14	7

Table 1: Confusion matrix for explaining recall and precision

Simply put, recall is the measure of how many of the actual values have been recovered, whereas precision is the measure of how many of the recovered values were correctly recovered.

For this research, precision and recall are used to compare the disjoint and overlapping areas between predicted and actual tracks. When considering recall in the spatial domain this way, this research defines it as follows:

$$Precision = \frac{Intersection}{Predicted\ area} \tag{3}$$

$$Recall = \frac{Intersection}{Actual\ area} \tag{4}$$

4 Methodology

In this section, the steps taken to answer the previously described research questions are detailed. This includes the chosen obfuscation methods, the applied attack strategies, and a method of evaluating the results. All relevant code can be found at `https://github.com/Martin13131/GIMA-thesis`.

4.1 Data

This research used the B-riders data-set (de Kruijf, 2014); this data-set consists of over 3.3 million individual points, divided into 39.000 unique tracks. All tracks come from 580 users with home locations included estimated according to the method in Feng and Timmermans (2017). In order to obtain road vertices for the crowding algorithm, the fietsersbond bicycle road network is also used as a reference.

4.2 Obfuscation methods

In order to keep the scope of this research feasible, three obfuscation methods were selected to implement and test. The chosen obfuscation methods are grid masking, random perturbation, and crowding. These have been chosen since grid masking and random perturbation are commonly used (Seidl et al., 2015), whereas crowding is a promising new obfuscation technique introduced by Scheider et al. (2019). More importantly, they provide three different types of approaches to obfuscation, as described below.

4.2.1 Grid masking

Grid masking is a prime example of GPS points being snapped to a pre-existing structure, and more applicable to track information than, e.g. voronoi masking. During the implementation, an average grid distance of 250 meters was used. This distance was chosen as previous research already indicated a distance of 200-350 m (Leitner and Curtis, 2006) was sufficient to distort the data so that its perceived similarity was substantially different, and Seidl et al. (2016) found this to be the most effective grid masking distance threshold overall.

4.2.2 Random perturbation

Random perturbation provides an offset-based obfuscation and subsequently adds a lot of randomness(Sila-Nowicka and Thakuriah, 2016). This makes it more difficult for a deterministic attack strategy to be applied. The main parameter for this type of algorithm is the offset amount, which was set at 10% of the upper and lower bounds of both the x and y directions per track.

This distance was chosen in order to maintain a relevant spatial distortion, since the area covered in a single track can vary wildly across different tracks, and a flat distance has a strong chance to not distort the track enough in tracks with a high distance to attain an accurate privacy measure, or too much for low distance tracks, which would then become unusable for our stated purpose.

4.2.3 Crowding

Finally, crowding does not simply alter the original data but instead adds additional points. This process can make it more difficult for an attacker to extract the relevant information from an obfuscated track. This research used a specific implementation of a crowding algorithm, described below.

In order to account for credible location and movement constraints, the road network in a buffer zone of 500 meters was taken. From this clipped road network, all vertices were added to a pool of possible locations. Subsequently, a maximum number of either 1000, or the total number of available vertices were randomly sampled from this pool of possible points, and added to the original track. This number of 1000 available points was chosen based on a test-sample of different tracks and visual inspection showed that 1000 points are sufficient to

cover the majority of the track, whereas 500 added points still made it relatively easy for the track to be discerned visually.

4.3 Evaluation

Before covering possible attack strategies, it is important to consider the purpose of the intended attacker, and how to evaluate an attack strategy once it has been applied.

Thinking about possible attacks, there are different purposes possible for someone wanting to attack data obfuscations. This could be trying to find a person's home address, finding which route they usually take, or completely reconstructing the original track point by point for other, possibly analytical purposes. These goals would all require a different evaluation metric on which to score an attack strategy. This research proposes three different attack strategy evaluation metrics, depending on the intended purpose.

Firstly, corresponding to the goal of finding a person's home location, the loss function must reflect the distance of the actual home location to the home location proposed by the attack strategy. This can further be normalized based on several factors such as the total length of the track, or other measures, but in essence, the distance between these two points is the key factor.

A problem with this approach is that tracks do not necessarily contain the home location, as the route between someone's work and e.g. the supermarket can also be considered a single track. Naturally, this can also provide information about the user's identity, as demonstrated by Fechner and Kray (2012), but this still requires human deduction as an attack strategy, and is therefore not applied in this research.

Secondly, when an attacker tries to reconstruct the original route a person took, it is not relevant if each point is at the right location, as long as the shape of the track remains the same. In order to obtain this shape, a set of points is not representative of the original track, so they will need to be converted into line features. From line features, we can then calculate the distance between the line of the actual track, and the line of the proposed track from an attack strategy.

However, when using line features to compute our loss function, the only metrics available to us are the distance between two lines or the overlap of the two lines. Thus, in order to obtain different metrics, a buffer is added to convert the lines into polygons. This has the advantage of providing overlap and a possibility to use metrics like precision, recall and intersection over union (IoU), also referred to as the Jaccard index. These can be computed by calculating both the intersection and the union of the actual and predicted polygons, and result in a standardized, meaningful score.

Furthermore, this approach will discount possible small errors that may be caused by to circumstantial factors such as GPS errors, as the polygonal overlap will still occur when a prediction is slightly off, which would not occur with e.g. line-based evaluation.

For this research, the evaluation measure using polygonal overlap has been used to evaluate the results of different attack strategies. While the home location estimation is the most likely evaluation measure an attacker would use, as this is the most sensitive information, this would require points of interest to be marked as temporary home locations from which a user profile could then be built up. The data used in this research only has home locations marked, and is therefore unsuitable for testing attack strategies with this purpose. By using the polygonal overlap method, the most information as to the functioning of each attack strategy can be extracted, making it the most suitable version. In this research, a buffer of 10 meters was used to generate these polygons, as this corresponded to the GPS-error of the tracks.

As the polygonal overlap evaluation metric is computationally expensive, it was infeasible to apply this to the deep learning approach as its loss function, which is essentially the function it is optimising for. This is because each track is fed into the neural network up to hundreds of times, and computing the polygonal overlap measure would either require a significant portion of time through rewriting all required functions in a format suitable for GPU processing, or significantly increasing the already high training times of a neural network. Therefore, a different measure was used as a loss function in the training of the neural network.

For this purpose, the root mean squared error (rmse) function was applied on a point-by-point basis, effectively computing the distance between where a point is predicted, and where its counterpart in the actual track is located, and optimising to minimize this difference. This measure was chosen as it required a significantly lower computational investment, and due to its close relation to the geospatial domain.

4.4 Attack Strategies

As mentioned before, this research will take a red teaming approach to obfuscation algorithms. In that context, the generated train and test set can be seen as a malevolent outsider having obtained some obfuscated information from the server, as well as having an unobfuscated set of track for him to experiment with. This unobfuscated training set can be obfuscated using the provided service and examined for alterations in the data, while the test set is the intended target.

This research proposes firstly takes on a heuristic approach, applying manually specified methods to ascertain the location of the original data. Secondly, this research investigates the possibilities of using deep learning to ascertain the original track locations from obfuscated tracks.

4.4.1 Heuristic methods

Assuming no initial knowledge whatsoever of the user's actual location, there are several easy ways of narrowing down the possibility space. Speaking in terms of precision and recall, assuming no knowledge of the user's actual location implies

a possible area that covers the entire world. This has the maximum possible recall, since every point from a GPS track will be contained within this area. The goal of this approach, therefore, is to maximize our precision, with the minimum possible sacrifice to our recall value.

Bounding box

Firstly, and most reliably, simply taking the bounding box of the obfuscated track will provide a good starting point, since the original track will almost definitely be within the boundaries of the obfuscated track. Either that, or the original track must have been obfuscated so heavily there is almost no relation anymore between the obfuscated and the actual track. Using this approach, we still maintain a near maximum recall, while increasing our precision from an infinite possibility space.

Buffer approach

Narrowing this down further, a buffer can be generated from all points. The size of this buffer should be based on the difference between actual and obfuscated points in a return track. Since an attacker would be able to experiment with the underlying obfuscation algorithm by uploading points to the server, this metric would be relatively easy to obtain. In this research, the buffer approach has been implemented by generating a buffer equal in radius to the average distance between 2 consecutive points in the obfuscated track.

Although the previously mentioned algorithms are likely to have a high recall value, they are still based on a narrowing-down approach of where the data must be, and the precision will likely still be very low. However, the low required investment of this attack strategy, both in computation as well as in complexity, can be worthwhile for an initial attempt.

Sliding window approach

An approach that is likely to work on both the grid masking and the random perturbation algorithms is to apply a moving average on both the X and the Y values. Applied to grid masking, this will put points in between grid points to a possible location, recreating a new track somewhat similar to the actual track.

Applying this sliding window approach to the random perturbation obfuscation algorithm, this is likely to partially cancel out the generated noise. Since moving averages are generally used to smooth out noisy time series data (Alessio et al., 2002).

4.4.2 Deep learning

This research investigated the possibilities of applying deep learning methodologies as an attack strategy to obtain the original points from obfuscated tracks. Using deep learning, a data-set was generated with the obfuscated tracks as an

input and the original tracks as output. This was then split up into a training and a test set for training and validation, respectively. This research tried different models to test for better results. This research investigated two types of models to apply to this data.

The first type of model used was a recurrent neural network. Recurrent neural networks are commonly applied to time series data(Connor et al., 1994) and are also used for speech processing(Graves et al., 2013). Since track data has similar properties to time series, as they are a sequence of points with a commonality that need not necessarily be a fixed pattern, and since their variable length is similar to the problems encountered in natural language processing, this was deemed the most logical first approach.

Given the high training time of neural networks in general, and the GRU layer's similar capacity to LSTM's but with a faster training time, GRU layers were used for this recurrent neural network approach.

The second type of model that was used was a 1-dimensional convolution model. Since the sliding window approach effectively is a small 1D convolution, it stands to reason that many different convolutions with parameters estimated through a machine learning approach would have the capacity to perform significantly better. This rationale is also based on the recent advances in computer vision applications since 2D convolutional filters were no longer manually specified (Lim, 1990; Krizhevsky et al., 2012), but rather learned through similar deep learning models. Nowadays, almost all computer vision applications use deep learning to great results.

For this research, the recurrent neural network models were instantiated with an initial GRU layer, followed by five consecutive combinations of a GRU layer and a dense layer with an Exponential Linear Unit (elu) as the activation function. These layers all had 64 neurons each. Finally, the output layer was defined as a single dense layer with two neurons to get both an x and a y output.

The 1-dimensional convolutional model was specified similarly with an initial 1D convolutional layer, followed by 5 combinations of a 1D convolutional layer and a dense layer with the elu activation function of 64 neurons each, with a dense layer of two neurons as an output layer. For this model, the filter-size for each convolution was set at five.

Training both these models, a set of 10.000 randomly selected tracks were given as input, with 10% of this being used as a test set. As neural networks need a static input shape, each track was padded with zeros to equalize their lengths. Furthermore, each track was scaled to a mean of zero by subtracting the mean value of the x and y coordinates in each obfuscated track, as feature scaling has been shown to improve model results in deep learning (Taylor et al., 2018).

For the training measure, the rmse loss function was used. During the initial testing phase, this caused the models to only output zeros due to the large number of zeroes required for padding. Therefore, the loss function was modified to not take zero values into account.

The learning rate was set at an initial value of 0.01, multiplied by 0.9 after three consecutive epochs without an improvement in the training loss. The

choice to optimize for the training loss instead of the validation loss was made as the test error of the models coincided with the training error during the initial experiments, and overfitting was less of a problem than the actual model learning capacity.

5 Results

This section will detail the results from applying each attack strategy to the set of three selected obfuscation algorithms. As mentioned above, the metrics of precision, recall and IoU were obtained by creating polygons out of the original tracks with a buffer of 10 meters, corresponding to the average GPS-error, and comparing the overlap between the predicted area and the actual area. In the case where an attack strategy returned points instead of an area, such as in the moving average attack, the polygons were also generated by applying a 10-meter buffer on the predicted points.

In order to effectively compare the results, it is important to establish a baseline. While this baseline in terms of attack strategies is set by the bounding box algorithm, this research also checked the evaluation metrics on the pure obfuscated data, given below.

Obfuscation algorithm	Precision	Recall	IoU
Grid masking	0.1227	0.1359	0.0689
Random Perturbation	0.0189	0.2085	0.0176
Crowding	0.0249	0.6871	0.0249

Table 2: Evaluation scores for each obfuscation algorithm without applied attack strategy

5.1 Heuristic methods

In obtaining the results, the evaluation metrics were generated by applying each attack strategy and the proposed evaluation on the entire data-set of 37.000 tracks. Additionally, a selection of the results was exported into ArcGIS for closer inspection, which is also included in this section.

5.1.1 Bounding box

As is to be expected, the bounding box algorithm has a high recall value with a low precision for all obfuscation algorithms. The low precision can easily be explained by the large area that is "predicted", which would never be exactly the same as the entire track.

The recall scores, although high, increase with each algorithm. This is logical, as the grid masking algorithm snaps each point in the original track to its closest point on the grid, thus making it possible for a point to be snapped into

the bounding box, generating a slightly lower recall score. The random perturbation algorithm moves all points by a semi-randomly chosen distance, allowing for points to fall outside of the original track's domain, and thus increase the number of points captured within the bounding box. Finally, the crowding algorithm adds many points to the original track, only increasing the scope of the bounding box, while still maintaining the original track's area.

Comparing these scores to the generated baseline, it is clear that, while it performs better on recall, the low precision scores make it an unlikely algorithm to choose for an attacker interested in determining the exact area where a user has travelled.

Obfuscation algorithm	Precision	Recall	IoU
Grid masking	0.0062	0.9112	0.0062
Random Perturbation	0.0039	0.9960	0.0038
Crowding	0.0042	0.9999	0.0042

Table 3: Bounding box results for each obfuscation algorithm

Taking a sample track, these numbers are once again reflected in a geographic view. Each obfuscation method generates a progressively larger bounding box, and thus achieve increasing recall values and decreasing precision values. In this way, it is also clearly visible that the grid masking algorithm tends to decrease the domain of a track, whereas the random perturbation expands the possible domain, with the crowding algorithm providing the most significant increase in the total possibility space for the original track.

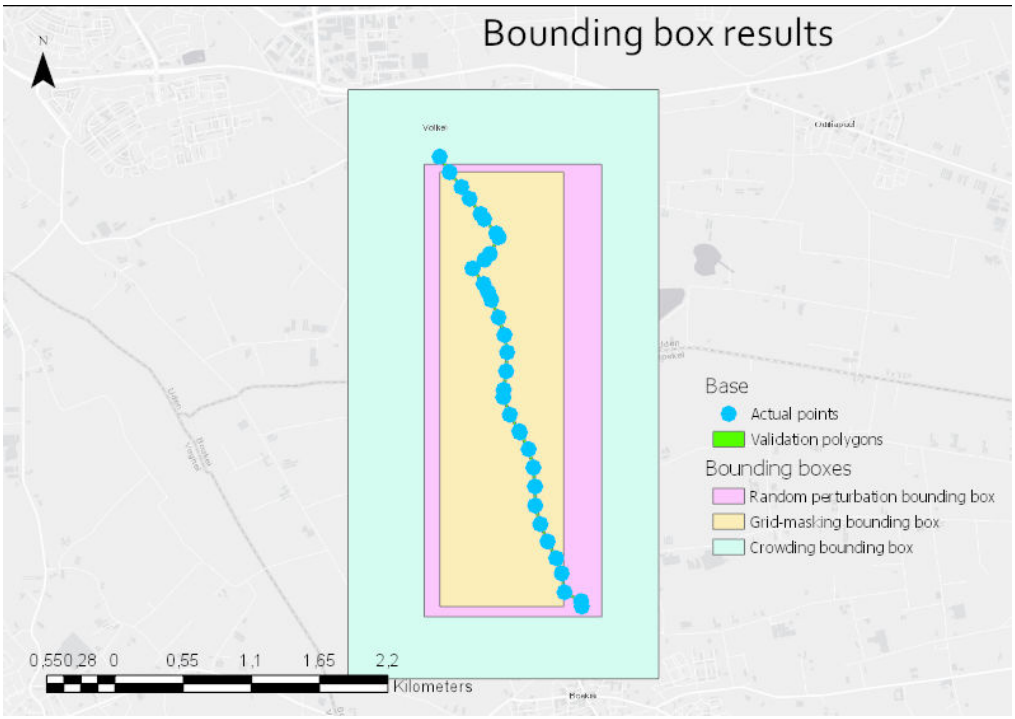


Figure 5: Bounding box results for a single track

5.1.2 Buffer approach

The buffer approach has a significantly higher precision score for grid masking than the bounding box algorithm, with a relatively minor reduction in recall. For crowding and random perturbation, there does not appear to be a significant change in the scores compared to the bounding box algorithm. Although they may be considered slightly better given the higher recall for random perturbation, or the higher precision for the crowding algorithm. Compared to the baseline, the recall is still significantly higher, although this attack strategy still sacrifices on precision.

Obfuscation algorithm	Precision	Recall	IoU
Grid masking	0.0638	0.7200	0.0622
Random Perturbation	0.0039	0.9978	0.0039
Crowding	0.0061	0.9967	0.0061

Table 4: Buffer attack results for each obfuscation algorithm

Looking at the geographic aspect of the buffer attack strategy, it is clearly

visible that the random perturbation algorithm has the lowest precision, as it covers a significantly larger area. The attack strategy appears to work best on the grid masking obfuscation, with a large part of the results covering the actual track.

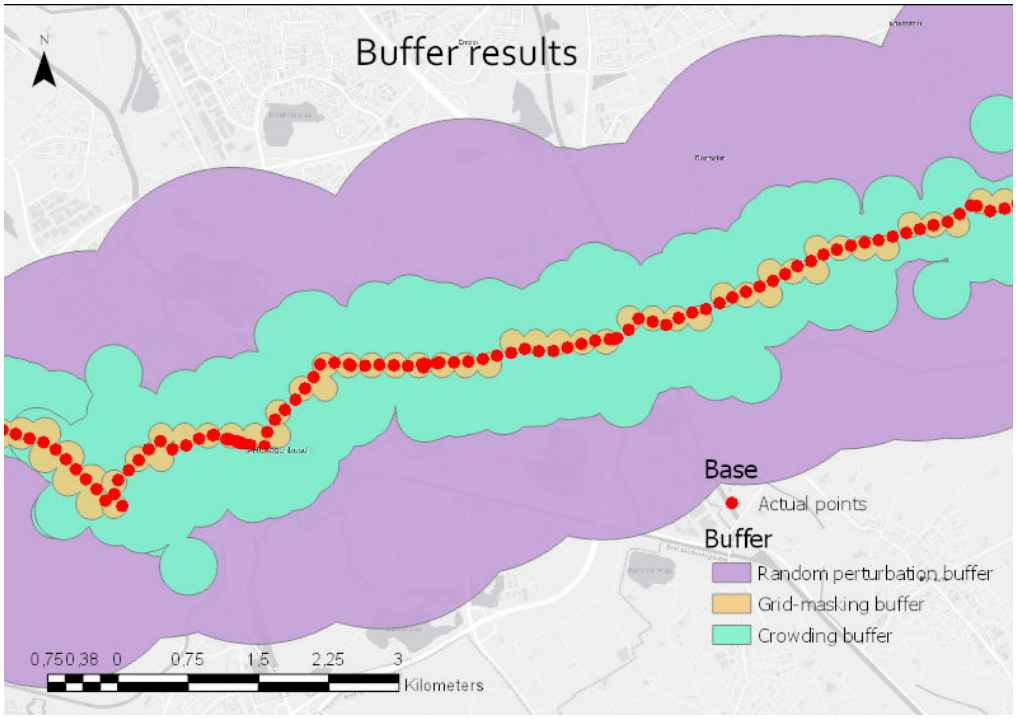


Figure 6: Buffer results for each obfuscation algorithm

5.1.3 Moving average

The moving average algorithm has a significantly lower recall than that of the other heuristic algorithms. There is, however, a slight increase in precision on the crowding obfuscation method compared to the other attacks. Comparing this with the baseline, the baseline is clearly better in terms of precision, and with the exception of grid masking, also on recall.

Obfuscation algorithm	Precision	Recall	IoU
Grid masking	0.0051	0.1710	0.0050
Random Perturbation	0.0033	0.1215	0.0033
Crowding	0.0090	0.4194	0.0089

Table 5: Moving average results for each obfuscation algorithm

The moving average, or smoothing attack strategy appears to behave differently depending on which obfuscation algorithm has been used. On the grid-masking algorithm, it appeared to work quite well, with the reconstructed track looking very similar to the original track, although the generated evaluation polygons are not overlapping enough to generate good metrics.

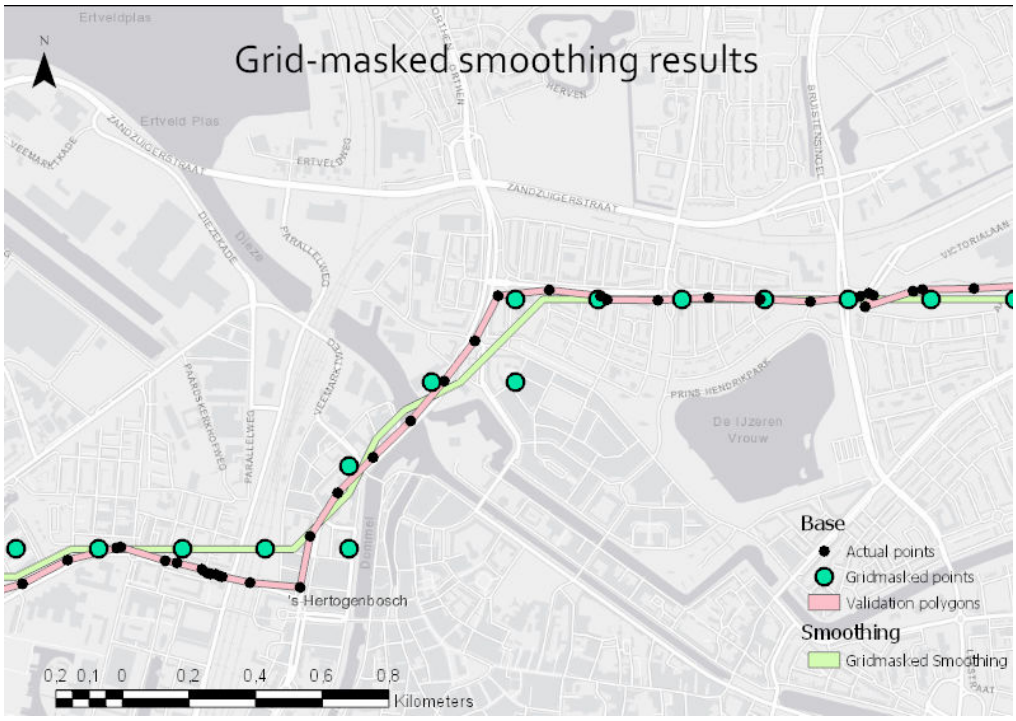


Figure 7: Moving average attack strategy on grid-masked data

Looking at the random perturbation results paints a significantly less pretty picture. It appears that, as a result of the random perturbation algorithm, points are intertwining in their track, with some points being perturbed to be further along in the track, distance-wise, while their sequential next points can get perturbed further back. The strategy in itself does show some promise, however as the three outliers in the upper left corner are clearly being mitigated by the rest of the points. This is indicated by the smoothing polygon being closer to the track than these outliers, and it stands to reason that with a more forceful reconstruction algorithm, this would better coincide with the original track's polygon. It is possible that e.g. with a higher window size, this strategy would produce better results.

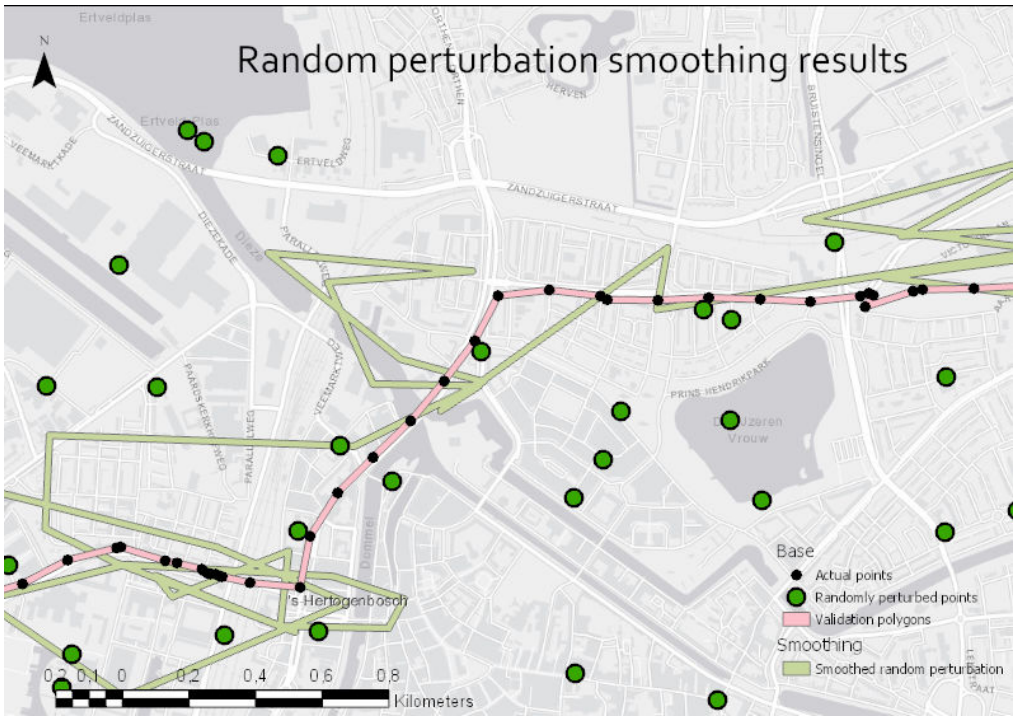


Figure 8: Moving average attack strategy on randomly perturbed data

Finally, looking at the crowding algorithm's results, these polygons appear to paint the perfect picture of a noisy time series. The generated polygon fluctuates around the validation polygon, overlapping at times but without a significant overlapping area to generate high metrics. Similar to the application to the random perturbation obfuscation, it is possible that a higher window size, or a higher weight placed on the surrounding elements would generate better results.

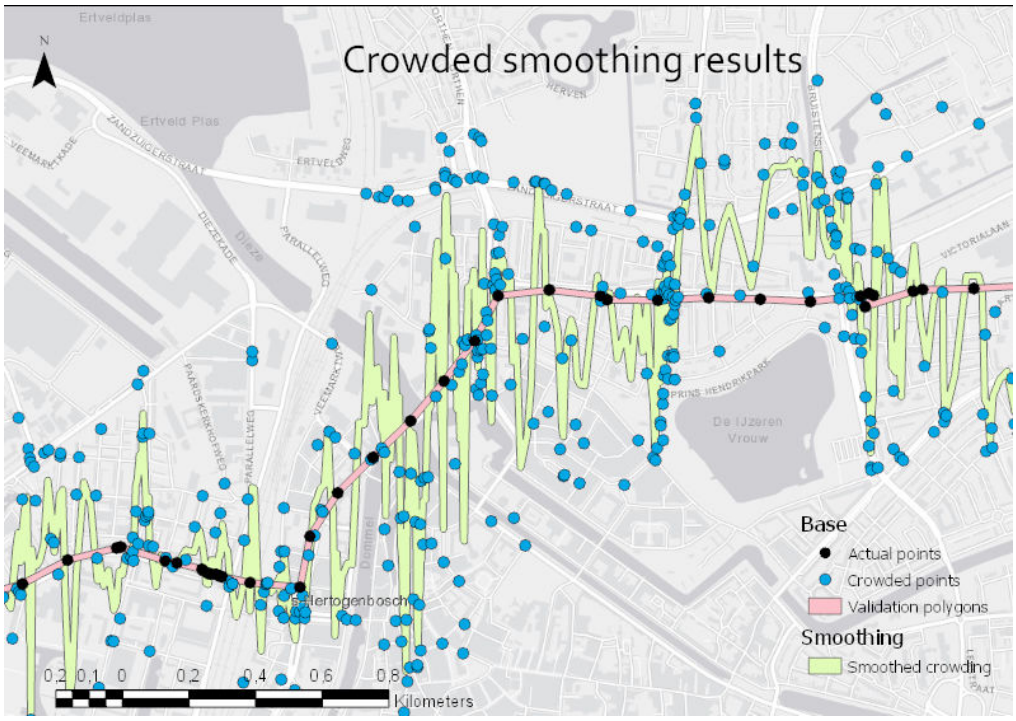


Figure 9: Moving average results for the crowding algorithm

5.2 Deep learning

For the evaluation, the models were applied to 27.000 tracks, with the 10.000 tracks that were used in training being excluded to ensure the results accurately measured the algorithm's performance, rather than it being a result of the model having been trained on that track.

Nearing the end of training time, all models appeared to plateau between rmse score between 2000 and 2500 on both the training and the test data. Translating this into distance, this equates to an average difference of 2-2.5 km between the actual and the predicted points.

Applying deep learning to the crowding algorithm in the specified configuration turned out not to provide any results. Given the custom loss function that was applied, masking all 0 values in the validation set for the model training, this effectively caused the neural networks to be blind to all crowded values, since these were not fed into the backpropagation algorithm as a result of this masking. However, the alternative of not using a loss function only resulted in zeros, as the applied sequence padding makes this the best probabilistic approximation of the validation data.

5.2.1 Recurrent neural networks

Applied to the grid masking and random perturbation algorithms, it appears there were no correct predictions. As all metrics returned a value of 0.0. Applied to the crowding algorithm, the results were as follows:

Obfuscation algorithm	Precision	Recall	IoU
Crowding	0.0385	7.0016e-05	6.9894e-05

Table 6: Recurrent neural network results for the crowding algorithm

Looking at the results from the convolutional networks, these results are slightly better, but by no means effective when compared to the heuristic strategies or even the baseline. Although the precision values are relatively high, this is possibly due to the large number of zeros added during the zero-padding in preprocessing.

Obfuscation algorithm	Precision	Recall	IoU
Grid masking	0.0494	0.0004	0.0004
Random Perturbation	0.0589	0.0.0004	0.0003
Crowding	0.0250	0.0002	0.0002

Table 7: Convolutional neural network results for both applied obfuscation algorithm

The results from the applied evaluation metric are also confirmed when looking at the results on a map. When focusing on a specific track, there is only the occasional area where a neural network prediction is visible, and even then it is not clearly related.

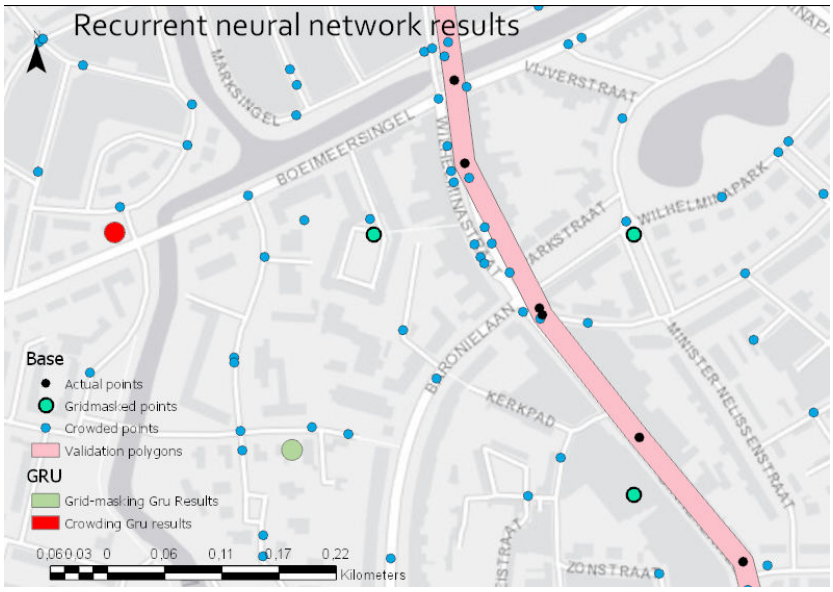


Figure 10: Results from the recurrent neural network approach

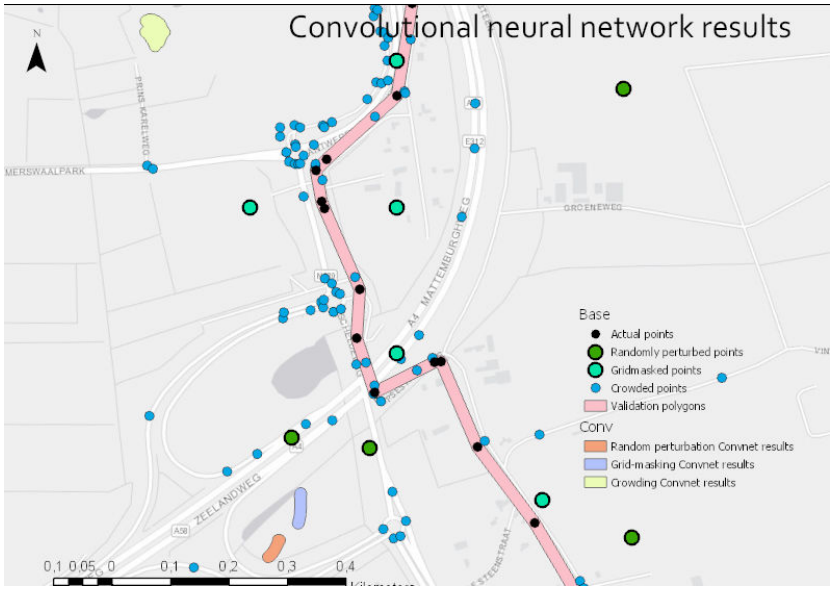


Figure 11: Results from the convolutional neural network approach

6 Discussion

This section will elaborate further on the implications of the results generated by this research.

6.1 Evaluation

Looking at the resulting polygons from the evaluation, we can see that lines that appear to coincide well with each other have low scores when the metrics are being computed. The prime example of this is the moving average results of the grid masked data. While these lines look very similar, and cross at times, this does not result in a high score for the evaluation. It should be considered to forgo the 10-meter buffer in a future research, to allow for a broader interpretation of when tracks overlap strongly enough.

Another interesting thing to note is the relatively high scores on both precision and recall the baseline seems to have. Because of this, taking the obfuscated points, creating a line and subsequently buffering by a certain margin may actually prove to be the most successful attack strategy this research produced.

A final important consideration related to the evaluation methods is spatial obfuscation research has been going on for over 16 years (Gruteser and Grunwald, 2003), and there is a distinct possibility that obfuscation algorithms have progressed to a point where it is very difficult to recover the original tracks from a modern spatial obfuscation method. If this is not the case, at the very least it takes a considerable amount of work to recover the original tracks, as demonstrated in this research.

6.2 Heuristic methods

Looking at the results of the heuristic attack strategies, it can be concluded that the moving average attack strategy does not function as intended, as it has the lowest precision and recall scores of all heuristic methods. The results do show there is room for improvement, and that the moving average algorithm has the potential to draw obfuscated points closer to their actual counterparts.

Looking at the buffer approach, on both the grid masking and crowding algorithms it performs better in terms of precision than the bounding box strategy, without sacrificing much in terms of recall. Especially applied to grid masking, it performs over ten times better in terms of precision, while only decreasing by 0.2 in terms of recall. Therefore, this strategy shows promise, especially if one were to apply a density function to get the center area of all buffered points.

For both these methods, it would be worthwhile to investigate the impact of their respective parameters i.e. window size and buffer size, respectively on their impact on the performance of the attack strategy.

6.3 Deep learning

Although the recall values are very low for the deep learning results, the precision values are significantly better than that of most heuristic attack strategies. This research was limited in model complexity as a result of the available computational resources, with a network of 5 layers deep and a width of 64 neurons per layer. Given the relatively high results on precision, it would be interesting to investigate neural networks with more capacity to apply to this problem.

Another point of contention is the rmse loss function that was applied for the network training loss. This effectively causes the model to optimize for a different metric than we are eventually scoring it on, which may have caused a disconnect between what the network trained to do, and what this research expected it to do. It may be worth investigating whether converting each track to its polygonal representation first, and rasterising it. This way, a 2D convolutional neural network could be applied for semantic segmentation, which could provide better results. This option is further discussed in section 8.2.2 of the recommendations.

6.3.1 Additional inputs

It should be noted that the deep learning approach applied in this research only gave the tracks as an input. While this produced some results, it is likely that these would be strongly improved by using additional inputs such as the surrounding road network. However, the embedding of a road network for deep learning requires a lot of thought as to how this can be well represented in the network. Bay and Sengupta (2017) have started research into this possibility, but would require more research to operationalize it as an addition to the deep learning attack strategy. This is further expanded on in section 8.2.1 of the recommendations.

6.4 Metrics

Looking at the results, one can see the precision and IoU corresponding strongly. While on first sight this may seem like an error, this co-occurrence can be explained by looking at the equations:

$$Precision = \frac{Intersection}{Predicted\ Area} \tag{5}$$

$$IoU = \frac{Intersection}{Predicted\ Area + Validation\ Area - Intersection} \tag{6}$$

When the predicted area is very large (causing precision to remain very low), the calculation for IoU will get skewed towards a precision measure, since the validation area and the intersection between the two are relatively small related to the actual predicted area. When precision is higher, the validation area and their intersection will play a larger role, which would make the IoU score more informative.

7 Conclusion

This research provides a first step in considering explicit, practical attack strategies. These have been divided into heuristic and deep learning approaches. For the heuristic approaches, the following algorithms were proposed:

- Bounding boxes
- Point buffers
- Moving average

Out of these approaches, this research shows that the buffer approach currently works best, but that the moving average attack strategy shows promise for further study. This research also experimented with using recurrent and convolutional neural networks to see if a machine learning approach could outperform a heuristic attack strategy, and clearly concludes that using the specified approach is not suitable for use as an attack strategy.

Although there are many as of yet unexplored possibilities for more advanced attack strategies which may perform better, the obfuscation algorithms examined in this research outperformed the proposed attack strategies. While there is still much research to be done into this topic, this research took a first step into the exploration of different attack strategies, so as to improve the security of modern spatial obfuscation methods.

8 Recommendations for future research

This research provided a first step in the study of attack strategies on obfuscated data. Throughout the duration of this research, choices were made to keep the scope manageable, but there are still many other avenues of attacking track data. This section will discuss the main ideas that are still unexplored, but which could provide further insights.

8.1 Heuristic methods

While this research proposed three heuristic methods to apply as an attack strategy, there are many different thinkable ways of computationally de-obfuscating track data. As mentioned in the discussion, using a heuristic strategy that is based on circumstantial data like road networks, land use, population density etc. could be applied. Another possibility is to apply a point density analysis and perform a cost-path on its result from the first to the last point in a track. While this fell beyond the scope of the current research, there are still many different attack strategies to create and evaluate.

8.1.1 Map matching

The research field of map matching, or mapping a set of points to their most likely location on a road network (Quddus et al., 2007), has many similarities

to the goal of this research of reconstructing a user's original track. The main difference is that this research considers obfuscated tracks, whereas a general map matching algorithm generally tries to eliminate the GPS error. Nevertheless, a map matching algorithm like the one mentioned in Marchal et al. (2005) could be worth investigating as an attack strategy, since the topics are so closely connected.

8.1.2 Reordering of points

The smoothing attack strategy as well as the deep learning attack strategy described in this research both used an approach which was reliant on the sequential ordering of points. When applying an obfuscation algorithm, it would then be an easy step to shuffle the points in a track to invalidate these attack strategies.

To compensate for this possibility, this research would suggest first applying a k-nearest neighbour algorithm, similar to the way it has been applied in the ISOMAP algorithm for use in multidimensional scaling (Tenenbaum et al., 2000). In this way, the most likely ordering of points can be recreated, after which the previously mentioned attack strategies can again be applied.

8.2 Deep learning

While this research investigated the preliminary possibilities of applying deep learning as an attack strategy, the vast majority of current deep learning research is looking into either computer vision (Badrinarayanan et al., 2015; Chen et al., 2017) or natural language processing (Deng et al., 2013; Amodei et al., 2016). While this research attempted to translate experiences from natural language processing to the field of privacy attack strategies, it was only possible to provide a first step. This subsection will contain recommendations for the further exploration of applying deep learning as an attack strategy.

8.2.1 Sequence to sequence models

For future research, it is suggested to apply sequence to sequence learning to the obfuscated GPS tracks. Sequence to sequence learning is a novel way of using LSTMs to transfer a sequence from one domain to another (Sutskever et al., 2014). Rather than passing the data through an LSTM layer and continuing on with the altered input data, they propose to use the in-memory representation of the LSTM as an encoder function. This representation is then transferred to another network, which then generates data based on this representation.

Such types of encoder-decoder models are used for many applications, such as machine translation (Cho et al., 2014), speech recognition (Bahdanau et al., 2016), and generating text responses in a chatbot-like setting (Shang et al., 2015). Applied to GPS-tracks, it stands to reason this could be used to build up a representation of the track, and subsequently generate a more abstract

version of the track in return. This could prove especially valuable as an attack strategy against the crowding algorithm.

A recent research by Bay and Sengupta (2017) was the first to apply a sequence to sequence model on geographic data. This was done by using Fisher embedding of the road network to embed it as an input to the neural network. After training, the model was able to generate the shortest path between two points on the network. While this operation in itself can be easily performed through a Dijkstra algorithm (Dijkstra, 1959), the novelty of this research lay in the geometric embedding of the road network for use in a neural network. Extrapolating on this possibility, this could be the start of a new branch of research into geographical vector data using deep learning, where one of the possibilities entails the use of such a network as an attack strategy.

8.2.2 Track rasterization

Since the field of computer vision has received so much attention in recent years, another possible approach to achieve better results for a deep learning attack strategy would be to rasterise the buffer of the actual tracks as a training set, and apply a convolutional segmentation model to the data. In this way, it would be possible to use a model that has been proven to provide good results in a different setting, such as e.g. the Tiramisu model (Jégou et al., 2017), and apply this as an attack strategy. Furthermore, this allows the model to use a metric such as intersection over union, which would correspond better with the goal of determining the travelled area, as opposed to the current approach where the model is effectively trained to determine the actual location of an obfuscated point, rather than the route taken on which the final evaluation is conducted.

8.2.3 Generative Adversarial Networks

A relatively recent development in neural networks is the concept of generative adversarial networks (GANs). GANs are made up of two networks, a generator network and a discriminative (classifying) network (Goodfellow et al., 2014). The generative network learns a distribution and generates data based on that distribution, whereas the discriminator compares it to actual data and classifies it as either being close to real data, or as significantly different from real-world data. This is where the adversarial part comes in; these two networks are directly in competition with one another, where the generator tries to "fool" the discriminator into believing the newly generated data is actual data.

Applied to the current research, a GAN could be used both as an obfuscation method, where the generator creates points based on the distribution of the input track, and an attack strategy while the discriminator tries to break the obfuscation by being trained on recognising actual versus fake data. Failing to break the obfuscation would then result in a strong privacy preserving algorithm. While it has not yet been applied in the context of privacy and attack strategies, it is certainly an interesting possibility.

References

Alessio, E., Carbone, A., Castelli, G., and Frappietro, V. Second-order moving average and scaling of stochastic time series. *The European Physical Journal B-Condensed Matter and Complex Systems*, 27(2):197–200, 2002.

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning*, pages 173–182, 2016.

Badrinarayanan, V., Kendall, A., and Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015.

Bahdanau, D., Chorowski, J., Serdyuk, D., Brakel, P., and Bengio, Y. End-to-end attention-based large vocabulary speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4945–4949. IEEE, 2016.

Bay, A. and Sengupta, B. Geoseq2seq: Information geometric sequence-to-sequence networks. *arXiv preprint arXiv:1710.09363*, 2017.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Connor, J. T., Martin, R. D., and Atlas, L. E. Recurrent neural networks and robust time series prediction. *IEEE transactions on neural networks*, 5(2): 240–254, 1994.

de Kruijf, W. Bike print: bike policy renewal and innovation by means of tracking technology. 2014.

Deng, L., Li, J., Huang, J.-T., Yao, K., Yu, D., Seide, F., Seltzer, M. L., Zweig, G., He, X., Williams, J. D., et al. Recent advances in deep learning for speech research at microsoft. In *ICASSP*, volume 26, page 64, 2013.

Dijkstra, E. W. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

Fechner, T. and Kray, C. Attacking location privacy: exploring human strategies. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 95–98. ACM, 2012.

Feng, T. and Timmermans, H. J. Using recurrent spatio-temporal profiles in gps panel data for enhancing imputation of activity type. *Big Data for Regional Science*, 2017.

Gers, F. A., Schmidhuber, J., and Cummins, F. Learning to forget: Continual prediction with lstm. 1999.

Ghinita, G. Private queries and trajectory anonymization: a dual perspective on location privacy. 2009.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Graves, A., Mohamed, A.-r., and Hinton, G. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE, 2013.

Grossberg, S. Recurrent neural networks. *Scholarpedia*, 8(2):1888, 2013.

Gruteser, M. and Grunwald, D. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of the 1st international conference on Mobile systems, applications and services*, pages 31–42. ACM, 2003.

Gruteser, M. and Hoh, B. On the anonymity of periodic location samples. In *International Conference on Security in Pervasive Computing*, pages 179–192. Springer, 2005.

Hamilton, J. D. *Time series analysis*, volume 2. Princeton university press Princeton, NJ, 1994.

Haykin, S. and Network, N. A comprehensive foundation. *Neural networks*, 2 (2004):41, 2004.

Hern, A. Fitness tracking app strava gives away location of secret us army bases, Jan 2018. URL `https://www.theguardian.com/world/2018/jan/28/fitness-tracking-app-gives-away-location-of-secret-us-army-bases`.

Hu, Y., Janowicz, K., Carral, D., Scheider, S., Kuhn, W., Berg-Cross, G., Hitzler, P., Dean, M., and Kolas, D. A geo-ontology design pattern for semantic trajectories. In *International conference on spatial information theory*, pages 438–456. Springer, 2013.

Jégou, S., Drozdzal, M., Vazquez, D., Romero, A., and Bengio, Y. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1175–1183. IEEE, 2017.

Kaasinen, E. User needs for location-aware mobile services. *Personal and ubiquitous computing*, 7(1):70–79, 2003.

Kido, H., Yanagisawa, Y., and Satoh, T. An anonymous communication technique using dummies for location-based services. In *ICPS'05. Proceedings. International Conference on Pervasive Services, 2005.*, pages 88–97. IEEE, 2005.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Krumm, J. Inference attacks on location tracks. In *International Conference on Pervasive Computing*, pages 127–143. Springer, 2007.

Krumm, J. A survey of computational location privacy. *Personal and Ubiquitous Computing*, 13(6):391–399, 2009.

Kwan, M.-P., Casas, I., and Schmitz, B. Protection of geoprivacy and accuracy of spatial information: how effective are geographical masks? *Cartographica: The International Journal for Geographic Information and Geovisualization*, 39(2):15–28, 2004.

LeCun, Y., Bengio, Y., et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *nature*, 521(7553):436, 2015.

Leitner, M. and Curtis, A. A first step towards a framework for presenting the location of confidential point data on maps—results of an empirical perceptual study. *International Journal of Geographical Information Science*, 20(7):813–822, 2006.

Lim, J. S. Two-dimensional signal and image processing. *Englewood Cliffs, NJ, Prentice Hall, 1990, 710 p.*, 1990.

Mao, A., Harrison, C. G., and Dixon, T. H. Noise in gps coordinate time series. *Journal of Geophysical Research: Solid Earth*, 104(B2):2797–2816, 1999.

Marchal, F., Hackney, J., and Axhausen, K. Efficient map matching of large global positioning system data sets: Tests on speed-monitoring experiment in zürich. *Transportation Research Record: Journal of the Transportation Research Board*, (1935):93–100, 2005.

Neidell, M. J. Air pollution, health, and socio-economic status: the effect of outdoor air quality on childhood asthma. *Journal of health economics*, 23(6): 1209–1236, 2004.

Nolan, J. and Levesque, M. Hacking human: data-archaeology and surveillance in social networks. *ACM SIGGROUP Bulletin*, 25(2):33–37, 2005.

Padilla, C. M., Kihal-Talantikite, W., Vieira, V. M., Rossello, P., Le Nir, G., Zmirou-Navier, D., and Deguen, S. Air quality and social deprivation in four french metropolitan areas—a localized spatio-temporal environmental inequality analysis. *Environmental research*, 134:315–324, 2014.

Powers, D. M. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2011.

Quddus, M. A., Ochieng, W. Y., and Noland, R. B. Current map-matching algorithms for transport applications: State-of-the art and future research directions. *Transportation research part c: Emerging technologies*, 15(5):312–328, 2007.

Scheider, S., Wang, J., Karssenberg, D., and Schmitz, O. Obfuscating spatial point tracks with simulated crowding. in progress, 2019.

Seidl, D. E. *Striking the balance: Privacy and spatial pattern preservation in masked GPS data*. PhD thesis, San Diego State University, 2014.

Seidl, D. E., Paulus, G., Jankowski, P., and Regenfelder, M. Spatial obfuscation methods for privacy protection of household-level data. *Applied Geography*, 63:253–263, 2015.

Seidl, D. E., Jankowski, P., and Tsou, M.-H. Privacy and spatial pattern preservation in masked gps trajectory data. *International Journal of Geographical Information Science*, 30(4):785–800, 2016.

Shang, L., Lu, Z., and Li, H. Neural responding machine for short-text conversation. *arXiv preprint arXiv:1503.02364*, 2015.

Shokri, R., Theodorakopoulos, G., Le Boudec, J.-Y., and Hubaux, J.-P. Quantifying location privacy. In *Security and privacy (sp), 2011 ieee symposium on*, pages 247–262. IEEE, 2011.

Sila-Nowicka, K. and Thakuriah, P. The trade-off between privacy and geographic data resolution. a case of gps trajectories combined with the social survey results. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 535–542, 2016.

Sutskever, I., Vinyals, O., and Le, Q. V. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Sweeney, L. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

Taylor, B., Marco, V. S., Wolff, W., Elkhatib, Y., and Wang, Z. Adaptive deep learning model selection on embedded systems. In *Proceedings of the 19th ACM SIGPLAN/SIGBED International Conference on Languages, Compilers, and Tools for Embedded Systems*, pages 31–43. ACM, 2018.

Tenenbaum, J. B., De Silva, V., and Langford, J. C. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Thurm, S. and Kane, Y. I. Your apps are watching you. *The Wall Street Journal*, 17(1), 2010.

Werbos, P. J. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

Williams, S. D., Bock, Y., Fang, P., Jamason, P., Nikolaidis, R. M., Prawirodirdjo, L., Miller, M., and Johnson, D. J. Error analysis of continuous gps position time series. *Journal of Geophysical Research: Solid Earth*, 109 (B3), 2004.

Wood, B. J. and Duggan, R. A. Red teaming of advanced information assurance concepts. In *DARPA Information Survivability Conference and Exposition, 2000. DISCEX'00. Proceedings*, volume 2, pages 112–118. IEEE, 2000.

Wordsworth, M. 'stalker apps' and gps allow domestic violence abusers to discover hidden refuges. *ABC news*, Jun 2015. URL `http://www.abc.net.au/news/2015-06-28/stalker-apps-and-gps-endanger-domestic-violence-victims/6570882`.

Zandbergen, P. A. Ensuring confidentiality of geocoded health data: assessing geographic masking strategies for individual-level data. *Advances in medicine*, 2014, 2014.