Expert Systems With Applications 165 (2021) 114068

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Analysis and experimental evaluation of the Needleman-Wunsch algorithm for trajectory comparison

Maroš Čavojský [a,*], Martin Drozda [a], Zoltán Balogh [b]

[a] Faculty of Electrical Engineering and Information Technology, Slovak University of Technology, Bratislava, Slovakia
[b] Department of Informatics, Constantine the Philosopher University, Nitra, Slovakia

ARTICLE INFO

Keywords:
Needleman-Wunsch algorithm
GPS (Global positioning system)
Sequence alignment
String edit distance
User trajectory comparison and experimental evaluation

ABSTRACT

We evaluate whether the Needleman-Wunsch algorithm is suitable for user trajectory comparison. The problem that we aim to solve is pair-wise user trajectory comparison. Similar user trajectories are then clustered with respect to their similarity, where clusters emerge in a non-supervised way.

We assume that user position, provided by GPS (Global positioning system), is normally distributed around user actual position. This assumption allows us to derive a model for setting score for match, penalty for gap and penalty for mismatch, which are an input to the Needleman-Wunsch algorithm. Our model implies that, in scenarios where actual user position is unknown and must be thus estimated from measured positions, the Needleman-Wunsch algorithm may be prevented from applying mismatches. In an experimental evaluation, we apply two data sets that contain recorded user positions and we show that our approach based on the Needleman-Wunsch algorithm is capable of correct classification of user trajectories into groups. Unlike in existing literature, we show that in GPS based user trajectory comparison, it is indeed not necessary to consider mismatches when applying the Needleman-Wunsch algorithms. This leads to a simplified string editing problem known as Longest Common Subsequence (LCS). We compare our approach with Edit Distance on Real sequence (EDR) in order to provide an insight into the performance of our approach.

Applying the Needleman-Wunsch algorithm has helped to solve several problems that emerge in GPS based user trajectory comparison such as interrupted GPS service due to satellite occlusion and various signal propagation phenomena such as signal reflection, fading etc. In order to improve the efficiency of the Needleman-Wunsch algorithm, we apply Move ability to identify when such detrimental conditions could occur. We also apply linear approximation in order to enhance user GPS trajectories with missing points, what further improves the efficiency of user trajectory comparison.

1. Introduction

Evaluating trajectories of people, animals and/or objects is essential for understanding existing and emerging links between spatial and social structure. Applications of such evaluation range from vehicle navigation to social applications. In their seminal paper on social organization of ants, Mersch, Crespi, and Keller (2013) tracked ants tagged with a unique bar code and continuously recorded their whereabouts in ant colony with a high-resolution camera. The ability to track ant movement on an individual basis allowed for understanding how ants during their lifetime regularly change social groups and migrate to different jobs.

The results by Mersch et al. also show that evaluation of trajectories is a task with unexpected instances. Herein, we focus on comparing trajectories of people that carry a mobile phone and are thus "tagged" with their unique mobile phone identifier. Our ambition is to propose, analyze and evaluate an efficient approach that could be applied, for example, in scenarios aimed at elucidating whether similar or dissimilar trajectories imply changes in social structure.

When evaluating trajectories for similarity, we assume that we deal with a discrete trajectory based on GPS (Global Positioning System) coordinates that are measured representations of a user's real position. Čavojský and Drozda (2019) observed that the discrepancy between a measured position and a real position can take form of a *nest* that is a result of various phenomena known to influence signal reception such as reflection, multi-path propagation or fading. Nests may occur when user

* Corresponding author.
E-mail addresses: maros.cavojsky@stuba.sk (M. Čavojský), martin.drozda@stuba.sk (M. Drozda), zbalogh@ukf.sk (Z. Balogh).

https://doi.org/10.1016/j.eswa.2020.114068
Received 8 May 2020; Received in revised form 25 August 2020; Accepted 27 September 2020
Available online 13 October 2020

is in an area where reflected GPS signal dominates. A nest can be viewed as a specific form of what is called *noise points* in Yang, Cai, Yang, Zhang, and Zhao (2020).

Gaps can have similar causes, additionally, they often emerge when a GPS device suffers a complete loss of signal, for example, due to satellite occlusion. Examples of these two phenomena are shown in Figs. 1(a) and (b). In controlled environments, such as the ant nests used by Mersch et al. in their experiments, gaps arise when the bar code attached to an ant becomes temporarily illegible.

1.1. Motivation

The existence of nests and gaps, connected with the reality that only estimates of a user's real position are available, motivates an approach for user trajectory comparison with the following properties:

- It is suitable for discrete user trajectories such as recorded by GPS.
- It can deal with nests and gaps, and as an extension with realistic environments with a high degree of noise, for example, due to various signal propagation phenomena.
- It is computationally feasible for trajectories that may be dominantly dissimilar and having significantly different lengths.

The last property makes the desired approach distinct from approaches aimed at comparing inherently similar sequences. For example, when comparing human DNA sequences it is reasonable to expect about 99.4% similarity (1000 Genomes Project Consortium, 2015), whereas in user trajectory scenarios such a high degree of similarity cannot be expected (unless nearly all users move alongside the same trajectory).

Approaches that are often considered for comparing trajectories include Euclidean distance, which may require trajectories of the same length with one-to-one correspondence between measured positions (Ranacher & Tzavella, 2014); longest common subsequence, which does not take into consideration size of gaps between common subsequences (Chen, Özsu, & Oria, 2005); nearest neighbor, which often leads to counting the number of nearby or identical positions in considered trajectories (Yuan & Raubal, 2012); discrete Fréchet distance, which is known to be sensitive to outliers (Ahn, Knauer, Scherfenberg, Schlipf, & Vigneron, 2010) and Dynamic time warping (DTW), which is aimed at cases when one trajectory is stretched or condensed version of the other sequence (Yuan & Raubal, 2012) and has therefore many applications in audio matching (Hu, Dannenberg, & Tzanetakis, 2003). Other approaches that are focused on identification of stay points (stop-overs) include (Yang et al., 2020), Move Ability (MA) is often used as an indicator to find stay points and noise points (Luo, Zheng, Xu, Fu, & Ren, 2017).

In Čavojský and Drozda (2019) the authors suggested that the Needleman-Wunsch algorithm (NWA) (Needleman & Wunsch, 1970), applied in alignment of DNA, RNA and protein sequences (and subsequences thereof), may also be suitable for user trajectory comparison. In bio-informatics, alignment of such sequences is necessary in order to argue about whether they could have a common ancestor with gaps emerging as a result of various evolutionary, structural and functional changes.

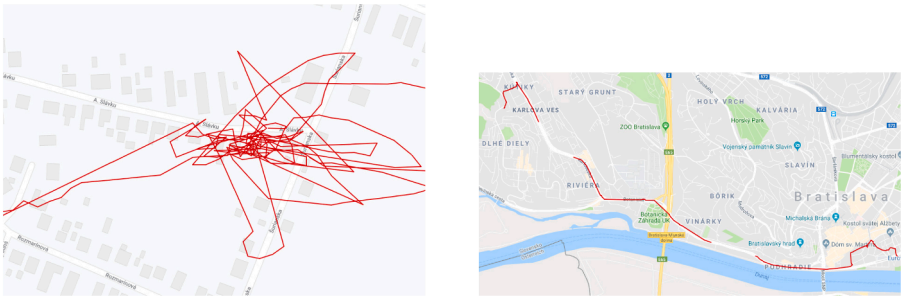
Even though, NWA is still widely used for optimal global alignment, several approaches that optimize or supersede NWA were proposed. Many of these approaches aim at massively parallel and distributed computing environments. To name a few, Chakraborty and Bandyopadhyay (2013) proposed a branch and bound approach for global pairwise sequence alignment. Vineetha, Biji, and Nair (2019) proposed an algorithm that takes advantage of the suffix tree for identifying common substrings. This results in improved NWA efficiency. The algorithm was implemented on an Apache Spark data framework in order to improve its scalability.

As the just discussed approaches are aimed at sequences of nucleotides, they do not offer any insights about how to deal with measured position sequences such as discussed herein. Additionally, these algorithms were proposed in order to deal with sequences having length of tens or even hundreds of millions nucleotides. Needless to say, sequences of measured positions applied herein are much shorter, however, these sequences show a higher degree of noise and dissimilarity.

1.2. Summary of results

The results presented herein, that reflect the above formulated goals, can be succinctly summarized as follow:

- We propose how to transform discrete user trajectories provided by GPS into a letter-based representation suitable as input to NWA. Such a representation requires that interpolated user positions are introduced in order to be able to compare two trajectories. We apply a linear interpolation between measured positions which may not be powerful enough to explain all user movements, however, we assume that users are free to move anywhere, they are not restricted to streets and sidewalks.
- We assume that the user's measured positions are normally distributed around the user's actual positions. We therefore model match and mismatch score for NWA as a function of normal distribution. By means of experimental evaluation we show that optimal classification results can be achieved.
- We show that it is not necessary to consider mismatches when applying NWA to user trajectory evaluation. This is a surprising result and it leads to a conclusion that an alignment computed by NWA needs to be further evaluated in order to argue about user trajectory similarity.
- NWA is an algorithm specifically designed for indentifying identical parts of sequences. In many applications including user trajectory comparison, parts of sequences that are dissimilar can provide valuable information. We apply Move ability to analyze dissimilar parts of sequences, so that nests can be effectively filtered out, two



(a) Nest - false movement. (b) Gap - missing user positions.

Fig. 1. Nest and gap.

trajectories that only differ in one or several nests can thus still be considered similar.

When preparing our experiments, we had to manually identify pairs of trajectories that were manually compared against a (street) map and declared to be either similar or dissimilar. In our experiments, we applied two data sets. The first data set is stemming from a separate long term experiment, where 455 mobile phones were distributed among students (Čavojský & Drozda, 2016). The advantage of this data set is that we could use its subset, where geographical positions of students were familiar to us and therefore easier to interpret. In order to avoid systemic bias, we also applied the Geolife data set by Microsoft Research Asia (Zheng, Fu, Xie, Ma, & Li, 2011). This data set recorded trajectories of 182 users with an abundance of similar trajectories.

The rest of this document is organized as follow. In Section 2 we discuss the related work. In Section 3 we introduce the Needleman-Wunsch algorithm and in Section 4 we discuss how this algorithm can be applied in trajectory comparison. In Section 5 we introduce EDR (Edit Distance on Real Sequence), an edit distance algorithm that we apply when discussing the quality and relevance of our results. In Section 6 we formally introduce the problem investigated herein, that is, evaluation of trajectory similarity. In Section 7 we introduce the experimental setup applied in our experimental evaluation. Section 8 contains the obtained results, and finally, in Section 9 we conclude and give suggestions for possible future work.

2. Related work

Čavojský and Drozda (2019) proposed an approach for comparing user trajectories based on NWA. They compared their approach with other approaches that are common when comparing user trajectory with possible trajectories derived from a street map (Yang, Zhang, Li, & Lian, 2011):

- Pairwise method that subsequently compares two positions having the same index i from each trajectory. This is only feasible for trajectories with equal length.
- Proximity method that compares two positions that are closest two each other.
- Upward proximity method that compares two positions that are closest to each other and that have not been used in comparisons yet. If a position has been already used, then the next position having a higher index will be applied.

They also demonstrated several challenges that need to be addressed, more specifically, nests and gaps that arise as a consequence of signal propagation effects and applied hardware.

Naidu and Narayanan (2016) applied NWA for identifying viral polymorphic malware variants. In order to apply NWA, they converted binary code to a fixed-size alphabet. They reported that they could detect several known viral polymorphic malware variants of JS.Cassandra virus and W32.Kitti virus, in some cases, with 100% accuracy. They also report that in some cases their approach has failed to deliver acceptable results.

Garhwal and Yan (2019) applied NWA for detecting watermarks in images. When encoding images, they first convert image to grayscale with 256 levels. Subsequently, they convert grayscale image to Byte64, hex format, binary format and finally to an alphabet of size four. For testing, they apply a standard image library with and without watermarks. For some data sets, they reported 100% accuracy. They show that watermarking results in areas that are similar to mutated regions in nucleotide sequences, hence the applicability of NWA in their scenario.

Ju, Park, Lim, Yun, and Heo (2018) investigated student smart card transactions and calculated similarity scores for finding relationship between students' trajectories and academic performance. They collected data for 685 students, computed standard t-tests for several groups and concluded that student daily trajectory is statistically significant for predicting academic performance. They applied a four-letter alphabet, where the letters corresponded to four locations: home, study, class and others.

Chua and Foo (2017) combine a decision tree classifier with NWA in order to recognize inhabitant's activity in a smart home. NWA is applied to compare the resulting decision tree models for similarity. They note that finding similarity in tree data structures has previously also been used for the analysis of XML documents.

Güyer, Atasoy, and Somyürek (2015) applied NWA to understanding navigation paths of users on the World Wide Web. Each web page was assigned a unique letter. The authors were comparing the optimal path for finding information to student (suboptimal) path. Their goal was twofold: (i) understanding the degree of disorientation with which a student is confronted and (ii) how disorientation could be mitigated.

When facing the problem of comparing sequences of eye movement, Day (2010) applied a bijective mapping of a small number of objects to letters in order to investigate user's eye fixation sequences. His experiment was realized in the context of a visually driven product selection process with applications to decision making.

Chen et al. introduced a distance function, Edit Distance on Real sequence (EDR), which they show is robust against noise, sensor failures or disturbance signals (Chen et al., 2005). They compared EDR with other approaches, most notably with Dynamic time warping (DTW), and conclude that DTW is sensitive to noise and is not suitable for comparing user trajectories.

Toohey and Duckham investigated the suitability of several similarity measures, including EDR and DTW, for comparison of trajectories of delivery drivers in the UK (Toohey & Duckham, 2015). Similar to the previous study, they also concluded that DTW is sensitive to outliers, however, they did not specify to what degree are outliers present in the applied data set.

Zheng gives an overview of trajectory data mining, indexing and retrieval, and reports that various filtering approaches such as median and mean filters are often applied for filtering outliers, i.e. for reducing noise (Zheng, 2015).

Taking into consideration the above reviewed related work, the relative lack of approaches for trajectory comparison that apply edit distance, with NWA being their instance, we decided to apply and evaluate NWA in settings where recorded user trajectory can be noisy due to signal propagation phenomena that lead to the aforementioned nests and gaps.

3. Needleman-Wunsch algorithm (NWA)

NWA is a prime example of dynamic programming, where by solving a series of smaller problems, it is possible to construct a global solution. It is also an instance of string editing algorithm, where the goal is to find minimum edit distance between two strings, that for example represent two sequences of DNA nucleotides:

Sequence 1:	G	T	C	G	A	C	G
Sequence 2:	G	A	T	T	A	C	A

An alignment for these two nucleotide sequences, when only identical letters are allowed to match, can be constructed as follows:

Sequence 1:	G	–	T	–	–	C	G	A	C	G
Sequence 2:	G	A	T	T	A	C	–	A	–	–

In the above example, the letters A (adenine), T (thymine), G (guanine) and C (cytosine) correspond to the four known types of DNA nucleotides, while "–" corresponds to introducing a gap. If we also allowed for mismatches, the alignment could be constructed as follows:

Sequence 1:	G	–	T	C	G	A	C	G
Sequence 2:	G	A	T	T	–	A	C	A

Mismatches model mutations, which can occur in DNA replication. A mutation can be a result of external conditions such as ultraviolet (UV) light, X-rays or various chemicals.

Unlike in the case of nucleotide sequences, when applying NWA to trajectory comparison, we face the additional challenge of applying a measured position instead of the user's real position. The precision of any measured position is related to the number of satellites with a fix that is influenced by signal propagation effects and available navigation data that can be either received from a satellite or over a network data connection. Imprecisely measured positions cannot be completely filtered out since, in general, we do not know user's real position. This implies that two trajectories, that are identical in reality, may show a degree of dissimilarity.

NWA takes as input two sequences, score matrix and gap penalty. The objective is to align these two sequences by matching letters and introducing gaps, where score with respect to matched letters and gap penalty for introduced gaps is computed. Matching only identical letters can be desirable, however, in bio-informatics score is often set according to mutation probabilities, where matching certain letters can have a higher score than matching other letters. It is also often desirable that alignment has few small gaps, therefore penalty for starting a gap can also be introduced.

NWA is a variant of string-editing algorithm, where the objective is to maximize alignment scores along the entire length of two sequences. Let $x_1, x_2, \ldots, x_m$ and $y_1, y_2, \ldots, y_n$ be two sequences, one having length m and the other length n. The scoring schema *score* defines scores when letters match or mismatch, for example in the case of nucleotide matching, $score(G, G) = 1$ and $score(G, T) = -1$, respectively. In its simplest form, *score* returns score 1 for identical letters and -1 for different letters including letter to gap mismatch (gap penalty). More complex schemes are often considered in order to capture different mutation probabilities; see e.g. BLOSUM scoring (Henikoff & Henikoff, 1992).

Having a scoring scheme, we can build a matrix M of size $(m+1) \times (n+1)$, where each entry $M(i,j)$ represents the score for optimal alignment of partial sequences $x_1, \ldots, x_i$ and $y_1, \ldots, y_j$. The matrix M needs to be initialized as follows: $M(0,0) = 0, M(i,0) = i * gp$ and $M(0,j) = j * gp$, where gp is gap penalty. The remaining entries of M are filled recursively:

$$M(i,j) = max \begin{cases} M(i-1, j-1) + score(i,j), \\ M(i-1, j) + score(i, \_), \\ M(i, j-1) + score(\_, j), \end{cases} \tag{1}$$

where $score(i, \_)$ and $score(\_, j)$ is the gap penalty equal to gp and $score(i, j)$ is the score of matching or mismatching at i-th and j-th position of sequences. In order to compute optimal alignment, we also need to record which of the three considered cases was applied, i.e. which resulted to the maximum value. In the case when identical value gets computed, a branching leading to several optimal alignments occurs.

When computing the optimum alignment we need to backtrack from $M(m,n)$ in the direction of recorded choices, where moving up or left means introducing a gap and moving on the diagonal means match or mismatch. NWA time and space complexity is $O(mn)$, therefore this algorithm is also suitable for computing sequence alignments of considerable length.

4. Mapping trajectory comparison to NWA

Since user trajectories consist of recorded positions and not letters, it is necessary to define the equivalence of positions. Let r_i and s_j be two recorded positions. r_i and s_j are equivalent if their mutual distance is less or equal ϵ:

$$|r_i - s_j| = \sqrt{(r_{i,x} - s_{j,x})^2 + (r_{i,y} - s_{j,y})^2} \leqslant \epsilon,$$

where $\epsilon \in \mathbb{R}_{\geq 0}, r_{i,x}$ and $s_{j,x}$ are x-coordinates of r_i and s_j, respectively, and $r_{i,y}$ and $s_{j,y}$ are y-coordinates of r_i and s_j, respectively. In other words, we consider Euclidean distance in two dimensions. A similar approach was taken by Chen et al. (2005), however, they applied Manhattan distance.

We implicitly apply an alphabet $\{R_1, R_2, \ldots, R_m\} \cup \{S_1, S_2, \ldots, S_n\} \cup \{X_1, X_2, \ldots, X_x\}$, where R_i denotes that the position belongs solely to Trajectory R, S_j denotes that the position belongs solely to Trajectory S and X_k denotes that the position belongs to both Trajectory R and S.

Fig. 2(a) shows an example with two intersecting trajectories. NWA could compute the following alignment:

Sequence 1:	–	–	R_1	R_2	X_1	–	R_3
Sequence 2:	S_1	S_2	–	–	X_1	S_3	–

Fig. 2(b) shows an example with two non-intersecting trajectories. NWA could compute the following alignment:

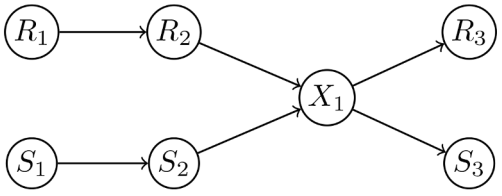
Sequence 1:	–	–	–	–	R_1	R_2	R_3	R_4
Sequence 2:	S_1	S_2	S_3	S_4	–	–	–	–

And finally, Fig. 2(c) shows an example with two similar trajectories. NWA could compute the following alignment:

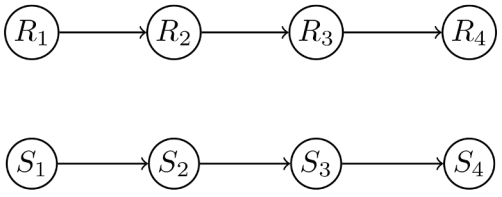
Sequence 1:	X_1	X_2	–	R_1	X_3
Sequence 2:	X_1	X_2	S_1	–	X_3

NWA could also compute the following alignment, this time applying mismatch instead of gaps:

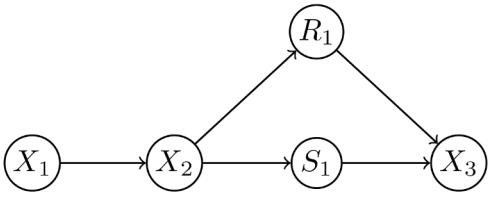
Sequence 1:	X_1	X_2	R_1	X_3
Sequence 2:	X_1	X_2	S_1	X_3



(a) Intersecting trajectories



(b) Non-intersecting trajectories



(c) Similar trajectories

Fig. 2. Examples of simple trajectories.

What alignment gets computed depends on the applied score matrix and gap penalty. Notice that in the case of two non-intersecting trajectories, the highest number of gaps gets introduced, however, if we set the cost for a mismatch low, then the alignment could be dramatically different.

In order to apply NWA we need to set *score* for matching and mismatching, as well as gap penalty *gp*. As we have already mentioned, often a simplistic scheme is applied with match score set to 1, and mismatch score and gap penalty set to -1. More complex schemes were proposed for efficient aligning of proteins, for example, BLOSUM scoring (Henikoff & Henikoff, 1992). This scoring takes into consideration the probabilities with which different proteins can be substituted. To calculate BLOSUM matrix, the following approach is applied:

$$score(i,j) = \frac{1}{\lambda}\log\left(\frac{p_{ij}}{q_i \cdot q_j}\right),$$

where p_{ij} is the probability that two aminoacids replace each other, q_i and q_j is the probability that aminoacid i and j, respectively, can be found in any protein sequence, and λ is a scaling factor, so that a suitable (rounded) integer values can be obtained.

The BLOSUM scoring suggests that when applying NWA in user trajectories comparison, a similar approach could be considered. The results by Heng, Gao, Walter, and Enge (2011) and Tiberius and Borre (2000) show that positions measured by GPS are normally distributed around user's actual position. This may depend on user environment, if considering an urban environment with buildings and other sources of occlusion, one might consider other models based on Rayleigh distribution or Rice distribution (Rappaport, 1996). We herein assume that we can model the distance of two positions as:

$$\phi(|r_i - s_j|) = \frac{1}{\epsilon\sqrt{2\pi}} e^{-\frac{1}{2\epsilon^2}|r_i - s_j|^2},$$

where r_i and s_j are measured positions of two users with actual position r_i^r and s_j^r, respectively, such that $r_i^r \equiv s_j^r$. We set match and mismatch score to be inversely proportional to the above equation:

$$match = \phi(|r_i - s_j|)^{-1} = \phi(0)^{-1} = \epsilon\sqrt{2\pi},$$

$$mismatch = -\phi(|r_i - s_j|)^{-1} = -\epsilon\sqrt{2\pi} e^{\frac{1}{2\epsilon^2}|r_i - s_j|^2},$$

where for *match* we assume that two positions r_i and s_j with $|r_i - s_j| \leqslant \epsilon$ are equivalent, i.e. we can substitute 0 for $|r_i - s_j|$. Since ϵ and $\sqrt{2\pi}$ scale *match* and *mismatch* proportionally, we can further simplify as well as apply the ceiling function:

$$match = 1,$$

$$mismatch = \left\lceil -e^{\frac{1}{2\epsilon^2}|r_i - s_j|^2} \right\rceil.$$

The rationale for setting gap penalty *gp* is that it is less than *match*, otherwise NWA would not align sequences, and equal or larger than *mismatch*, while having in mind that the probability that two positions belong to the same trajectory decreases with their relative distance:

$$match > gp \geqslant mismatch.$$

Given the above said, $|r_i - s_j| > \epsilon$ defines an area, where two positions are not equal, with *mismatch* also depending on ϵ:

$$\begin{aligned} &match = 1, \text{ if } |r_i - s_j| \leqslant \epsilon, \\ &mismatch = \left\lceil -e^{\frac{1}{2\epsilon^2}|r_i - s_j|^2} \right\rceil \leqslant -1, \text{ if } |r_i - s_j| > \epsilon. \end{aligned} \tag{2}$$

This defines how much freedom we have in choosing the various parameters, in one case two positions are nearby and we cannot distinguish between them due to GPS precision (or rather imprecision), in the other case, as their relative distance increases, their "mutation" chances decrease, i.e. chances that two distant locations belong to the same path decrease. In our experimental analysis, we therefore apply the following rule:

$$(match = 1) > (gp = 0) > mismatch = \{-1, \cdots, -k\}, \tag{3}$$

where $k \in \mathbb{Z}_{\geqslant 0}$.

4.1. NWA: formal algorithm description

Let us now formally define the steps described in the previous section. Let us assume that we wish to compare two sequences of GPS positions $r = (r_0, r_1, \cdots, r_p)$ and $s = (s_0, s_1, \cdots, s_q)$, where $p, q \in \mathbb{N}$. The rationale of the algorithm is, each time we apply a position belonging to either r or s, we look for the next position in r or s that is at least Δ meters further, where $\Delta \geqslant 2\epsilon$. We denote such a position as r_Δ and s_Δ, respectively. We use $\oplus$ to denote string concatenation. We assume that r_Δ and s_Δ implicitly define a label R_i and R_j, respectively. This way we build the strings R and S that are necessary as inputs to NWA. Unlike in the standard version of NWA that computes $M(i,0)$ and $M(0,j)$ before all remaining procedures, we compute $M(i,0)$ and $M(0,j)$ when needed. This approach implies that only a single pass of r and s is necessary.

The NWA variant that we apply is shown in Algorithm 1. The time complexity of this algorithm is $(m+1)(n+1) + \max\{m,n\} = O(mn)$. The sequences r and s require a single pass to convert them to a letter-based representation of length m and n, respectively. Each element of matrix M is only computed once, what requires $(m+1)(n+1)$ steps, and then backtracking is necessary to construct an alignment, what requires $\max\{m,n\}$ steps.

Algorithm 1. (*NWA for GPS sequence alignment*)

Require r, s
  $R \leftarrow r_0$
  $S \leftarrow s_0$
  initialize M ▷ compute $M(0,0), M(1,0), M(0,1)$
  while r not empty or s not empty
    recursively compute $M(i,j)$ ▷ see Eq. (1)
    if match or mismatch
      $R \leftarrow R \oplus r_\Delta$
      $S \leftarrow S \oplus s_\Delta$
      update $M(i,0)$
      update $M(0,j)$
    else ▷ gap is inserted into either R or S
      $R \leftarrow R \oplus r_\Delta$ or $S \leftarrow S \oplus s_\Delta$
      update $M(i,0)$ or $M(0,j)$
    end if
end while
compute alignment of R and S ▷ Backtracking

5. Edit distance on real sequence

Edit Distance on Real Sequence (EDR) was introduced by Chen et al. (2005) and we apply it as a fair comparison to NWA. Similar to NWA, EDR is a string editing approach that compares two sequences by computing the number of insertions and deletions (gaps in NWA), and substitutions (mismatches in NWA) that need to be applied to one sequence in order to become identical with the other sequence.

Let R and S be trajectories of lengths n and m, respectively. EDR is defined as follows:

$$EDR(R,S) = \begin{cases} n, \text{ if } m = 0, \\ m, \text{ if } n = 0, \\ \min\begin{cases} EDR(Rest(R), Rest(S)) + cost, \\ EDR(Rest(R), S) + 1, \\ EDR(R, Rest(S)) + 1, \text{otherwise}, \end{cases} \end{cases}$$

where $Rest(R)$ is the subtrajectory of R without the first element r_1:

$$Rest(R) = (r_2, r_3, \cdots, r_n),$$

and $cost = 0$, if and only if $|r_{i,x} - s_{j,x}| \leqslant \epsilon$ and $|r_{i,y} - s_{j,y}| \leqslant \epsilon$, and $cost = 1$ otherwise.

Similar to NWA, EDR can be used to compute an alignment of two sequences by recording applied choices. Unlike EDR, NWA applies a score matrix that can take different probabilities into account, whereas EDR only applies fixed costs. If insertion and deletion are not allowed, EDR becomes Hamming distance (sequences must have the same length). If substitutions are not allowed EDR becomes equivalent to Longest Common Subsequence (Navarro, 2001).

6. Problem formulation

Let $p(match), p(gap)$ and $p(mismatch)$ be probability with which match, mismatch and gap gets applied by NWA, respectively.

Our goal is to show that if score for *mismatch* is set according to Eq. (2) and ϵ is set to a reasonable (practical) value determined by Android horizontal accuracy then we can expect:

$$p(mismatch) \to 0. \tag{4}$$

This is caused by assuming in Eq. (2) that for $|r_i - s_j| > \epsilon$, we can distinguish between two positions, whereas this is not possible $|r_i - s_j| \leqslant \epsilon$ (due to GPS imprecision). This turns our problem to error Type 1 and Type 2 trade-off, which is an inherent implication of various signal propagation phenomena and GPS accuracy.

Another related goal is to show that in the case of *mismatch* score set according to Eq. (2), mismatches are not necessary for aligning sequences of positions, more specifically, they are not necessary for user trajectory comparison.

The consequence of $p(mismatch)$ being zero or nearing zero is that NWA becomes a variant of Longest Common Subsequence. In order to evaluate similarity of two sequences, it is necessary to consider their structure, rather than relying solely on the number of common positions. Therefore in our experimental evaluation, we also consider properties such as the number of subsequent gaps applied by NWA while aligning sequences.

In the light of the above said, when evaluating two trajectories on similarity, we define their similarity in terms of the number of identical positions and the number of subsequent gaps applied by NWA:

Definition 1. Trajectories R and S having length n and m, respectively, are similar, if the following holds:

$$\frac{\#match}{\max\{m,n\}} \geqslant \alpha,$$
$$max\_gap \leqslant \beta,$$

where $\#match$ is the number of matches, max_gap is the maximum number of subsequent gaps, both as computed by NWA for the two considered trajectories, $\alpha \in \mathbb{R} | 0.0 \leqslant \alpha \leqslant 1.0$ and $\beta \in \mathbb{Z}_+$.

Our evaluation of trajectory similarity is thus solely based on the number of matches and the number of gaps, what is sufficient when mismatches are impossible or improbable. What values the parameters α and β should take depends on the considered scenario. Herein we aim at trajectory similarity, therefore we consider values such as $\{0.75, 0.85\}$ for α and $\{3, 5, 10\}$ for β. Leaving β unbounded could be considered in scenarios where the goal is to detect nests (and stopovers in general) that are followed by a trajectory shared by several users.

In order to obtain comparable results, we evaluate both NWA and EDR applying the above equations. Sellers showed that approaches formulated in terms of maximizing similarity (NWA) and minimizing edit distance (EDR) are equivalent (Sellers, 1974). For that reason, we assume that they can be evaluated in the same way.

7. Experimental Setup

7.1. User position information

To define a movement of device (user), it is necessary to determine its position at any point. We define user GPS position and trajectory as follows:

Definition 2. Position P is a couple (x, y):

- x is latitude in decimal degrees (e.g. 48.1518568),
- y is longitude in decimal degrees (e.g. 17.0711559).

Definition 3. User trajectory r is a sequence of (GPS) positions $[(P_0, t_0), (P_1, t_1), \cdots, (P_p, t_p)]$, where t_i is the time when position P_i was recorded and p is the length of trajectory r.

Alongside device (user) position, other information such as horizontal position accuracy may be useful. According to the Android documentation (Google, 2020), horizontal accuracy A is defined as a radius with 68% reliability. In other words, if we draw a circle with radius A and center P, there is 68% probability (one standard deviation) that the actual position is inside of this circle.

7.2. Interpolation of trajectories

Let us consider the example depicted in Fig. 3(a). It shows two trajectories r and s that are identical, however, s has missing positions between S_s and S_e. A similar example is depicted in Fig. 3(c), however, this time s and r are not identical, even though as in the previous example they share positions R_s, S_s and R_e, S_e. The cases depicted in Figs. 3(a) and (c) could lead to the same alignment.

The difference between r and s can be explained in our case by applying a linear interpolation by which new nodes are added at distance ϵ on the line connecting R_s, S_s and R_e, S_e. This is depicted in Figs. 3 (b) and (d), it demonstrates that by applying linear interpolation, the two considered trajectories depicted in Figs. 3(a) and (c) become distinguishable for NWA.

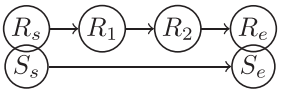
Applying linear interpolation does not mean that gaps will not be applied by NWA. A linear interpolation may not be powerful enough to explain all user movement, matching user movement to a street map could provide a more accurate interpolation, however, we herein assume that users are free to move anywhere and thus they are not restricted to streets and sidewalks. An example of originally recorded and interpolated trajectory is depicted in Figs. 4(a) and 4(b), respectively.

Other than interpolation, we did not apply any preprocessing in order to make data sets more suitable for analysis.

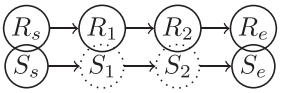
7.3. Data sets

The evaluation of our approach to sequence alignment is based on data sets that record user movement. We considered two data sets:

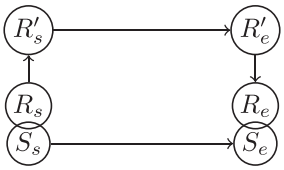
- The COhave data set was collected using 455 mobile devices distributed among students. It was collected during a 10-month period starting from September 2016 to July 2017. Over 20 million position records provide insights into students' behavior patterns (bars, restaurants, clubs etc.). Recording of positions was done using our implemented mobile application for energy efficient trajectory recording of mobile devices using WiFi scanning, described in more details in Čavojský and Drozda (2016) and Čavojský, Uhlar, Ivanis, Molnar, and Drozda (2018).
- The Geolife data set (Microsoft Research Asia) was collected by 182 users in a period of over three years (from April 2007 to August 2012). This data set contains 17,621 trajectories with a total distance



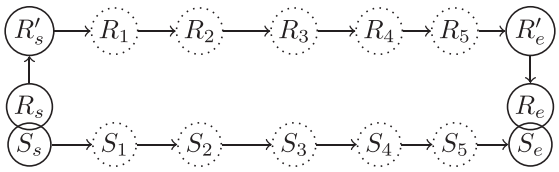
(a) Example 1: recorded positions. Trajectories r and s are identical. Intersecting nodes lie within ϵ distance.



(b) Example 1: recorded (solid line) and interpolated positions (dotted line).

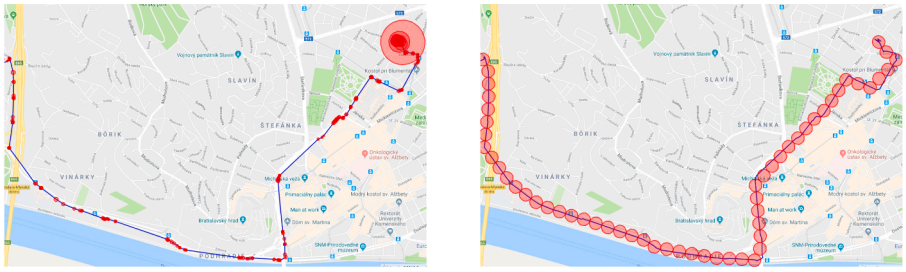


(c) Example 2: recorded positions. Trajectories r and s are not identical; R_s and S_s lie below (to the south) from R'_s, and R_e and S_e lie below (to the south) from R'_e. Intersecting nodes lie within ϵ distance.



(d) Example 2: recorded (solid line) and interpolated positions (dotted line).

Fig. 3. Interpolated trajectory.



(a) Recorded trajectory. (b) Interpolated trajectory.

Fig. 4. Applying linear interpolation.

of about 1.2 million kilometers and a total duration of 48,000 + hours. These trajectories were recorded by different GPS loggers and GPS capable phones, and have a variety of sampling rates. This data set recorded a broad range of users' outdoor movements, including not only life routines like go home and go to work but also some entertainment and sports activities, such as shopping, sightseeing, dining, hiking, and cycling (Zheng, Li, Chen, Xie, & Ma, 2008; Zheng, Zhang, Xie, & Ma, 2009; Zheng, Xie, & Ma, 2010).

These data sets contain user position information: unique user identifier, position, time when position is recorded and, if available, horizontal accuracy.

Tables 1 and 2 show the number of groups for the COhave and Geolife data set, respectively, each group containing a certain number of similar trajectories. Their similarity was evaluated by matching trajectories with a street map.

7.4. Move ability

Move ability is a concept introduced in Luo et al. (2017). Its purpose is to detect noisy areas in a GPS position sequence. It is based on comparing the Euclidean distance of its end points and the sum of

Table 1
COhave: the number of groups and trajectories.

Group	#trajectory
1	23
2	13
3	4
4	7
5	2
6	20
$\sum$	69

Table 2
Geolife: the number of groups and trajectories.

Group	#trajectory
1	118
2	26
3	13
4	269
5	24
6	8
7	1
8	1
$\sum$	460

distances of each successive pair of GPS positions.

Definition 4. Let $r = (r_0, r_1, \cdots, r_p)$ be a sequence of GPS positions. Move ability MA is then computed as:

$$MA = \frac{|r_0 - r_p|}{\sum_{i=0}^{p-1} |r_i - r_{i+1}|},$$

where $p \in \mathbb{Z}_+$. As this definition implies, the value of MA is smaller for sequences that are less direct, with some sideway movement, if compared to a straight line drawn between start and end position. The value 1.0 means that the sequence r represents a straight line. In our experiments, we assume that $r_0 \neq r_p$, i.e. user subtrajectory does not form a cycle, where $|r_0 - r_p| = 0$. In order to avoid $|r_0 - r_p| \approx 0$, if $|r_0 - r_p| < \epsilon$, we set the distance of r_0 and r_p to $\min(\epsilon, \sum_{i=0}^{p-1} |r_i - r_{i+1}|)$.

As per results presented in Luo et al. (2017), we chose to apply the value 0.5 as the threshold for deciding whether a subsequence could form a nest:

- $MA \in (0.0, 0.5]$ means that the subsequence is a nest,
- $MA \in (0.5, 1.0]$ means that the subsequence is not a nest.

We apply move ability as post-processing. For a subtrajectory, where NWA yields one or more gaps as a result, we compute its move ability. We thus try to estimate whether the gaps in this subtrajectory are caused by a nest (noise). If we confirm that the subtrajectory could represent a nest, we exclude this subtrajectory when computing similarity as defined in Eq. (1). More specifically, we do not add these gaps when computing *max_gap*. We report experimental results for both move ability applied and not applied.

Fig. 5(a) shows an example of two trajectories without move ability being applied. These two trajectories would be evaluated as not similar because the three nests lead to a large number of gaps being introduced by NWA and thus violating the threshold set by parameter β in Def. 1. Fig. 5(b) shows the same two trajectories with move ability applied. In this case, these two trajectories can still be evaluated as similar.

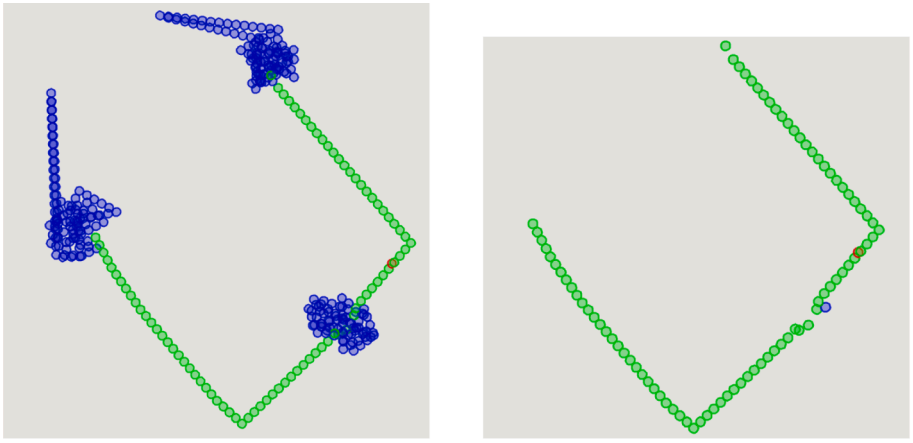
In Yang et al. (2020) the authors suggested that any trajectory T consists of either move points *MP*, stay points *SP* or noise points *NP*. The difference between *SP* and *MP* is that the activity behind *SP* can be interpreted as "place visit", whereas *NP* can be a result of various detrimental conditions such as satellite occlusion etc. They further suggested that an efficient approach to compute *NP* is to compute move points and stay points, where *NP* is the complement, i.e. $NP = T \setminus (MP \cup SP)$, where $MP \cap SP \cap NP = \emptyset$.

When applying NWA, we can assume that matches correspond to move points and gaps correspond to either disjoint trajectories, stay points or noise points. Move ability allows for distinguishing among these three cases, where disjoint trajectories and stay points are expected to have a high MA and nests (noise points) are expected to have a low MA. As NWA is an algorithm specifically designed for alignment computation, it offers an efficient way for finding similar parts of trajectories, dissimilar segments of trajectories need to be further analyzed with other tools, which may be application (or even scenario) specific. We therefore investigate whether MA is a suitable tool for analyzing dissimilar parts of trajectories, so that noise points (nests) can be filtered out.

8. Experimental Evaluation

The objectives of our experimental evaluation can be summarized as follow:

- Experimental verification that Eq. (4) holds for a range of ϵ values, while also considering several possibilities for the values of match, gap and mismatch, with focus on values that comply with Eq. (4),



(a) Without move ability. (b) With move ability.

Fig. 5. Nest detection with move ability.

- we can achieve a reasonable classification accuracy of NWA when applied to COhave and Geolife data sets, and that
- Move ability is further capable of improving the classification accuracy of results obtained with NWA.

We view the above objectives as necessary to show that NWA is a viable solution for user trajectory classification, especially since this algorithm has only been for this purpose previously evaluated in Čavojský and Drozda (2019).

8.1. NWA parameters

The parameters for NWA are set so that values of *match*, *gap* and *mismatch* follow the criteria set in Eqs. (2) and (3). In order to explore the parameter space, we contrast these value with several other possibilities.

The distance parameter ϵ was set to $\{2, 5, 10, 15, 20, 50, 80, 100, 150\}$ meters. The similarity parameter α was set to $\{0.75, 0.85\}$ and the similarity parameter β was set to $\{3, 5, 10\}$; see Def. 1.

8.2. Experimental results

In order to group user trajectories by similarity, we apply an approach similar to X-means clustering. A trajectory is either similar to a trajectory already assigned to a group, or this trajectory starts a new group. Tables 1 and 2 show that the number of groups in the COhave and the Geolife set is 6 and 8, respectively.

Fig. 6(a) and (b) provide visual help on how trajectories get compared. Fig. 6(a) shows two trajectories that get interpolated, so that NWA can get applied, and Fig. 6(b) shows the same two trajectories after NWA computed an alignment, where green color indicates matches, i.e. in those parts these two trajectories are identical.

Tables 3 and 4 show the classification results for NWA and EDR, for the Geolife and COhave data sets, respectively. The optimum result is shown with $*$. The tables show that indeed when applying Eqs. (2) and (3), the optimum classification result can be achieved, for the both considered data sets. Other values for NWA parameters could not deliver optimum classification results.

Tables 3 and 4 also show that the classification results for NWA and EDR, when Move ability is applied. We can see that computing Move ability helps improve classification performance for lower ϵ values, however, it dominates only in one case.

The tables also show that EDR performs better on the Geolife data set than on the COhave data set. EDR however perform worse than NWA with parameters set according to Eqs. (2) and (3).

The best classification results are obtained by setting ϵ to 50 meters. The horizontal accuracy of GPS devices, as estimated by Android OS, captured in the COhave data set is 10.37 ± 8.02 meter. The Geolife data set does not include information about horizontal accuracy. For two identical GPS positions this may mean that their real positions are $2 \times 18.39 = 36.78$ meters apart, what suggests why $\epsilon = 20$ meters delivers worse results than $\epsilon = 50$ meters, and why a further increase of ϵ is not helpful.

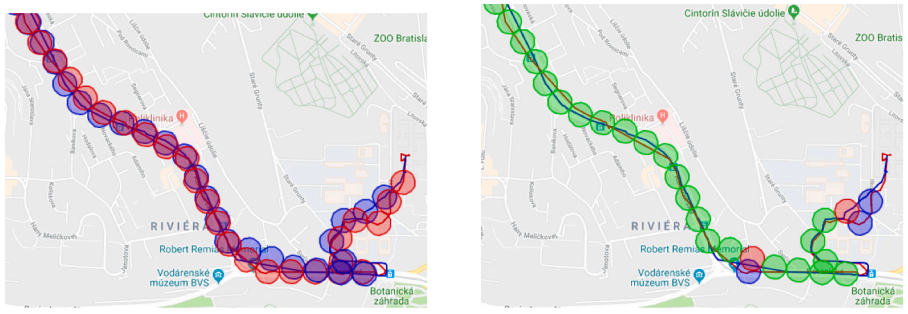
Table 5 shows for a very broad range of ϵ values that we can indeed expect Eq. (4) to hold. We applied two alternatives for $(match, gap, mismatch)$ values, $(1, 0, -1)$, which comply with Eqs. (2) and (3), and $(1, -1, 0)$, which do not. We can observe that in the former case the

Table 3
NWA vs. EDR: classification results on COhave dataset. $*$ indicates correct classification. $\alpha = 0.75$ and $\beta = 3$. (MA) indicates results with Move ability.

	ϵ [m]				
	20	50	80	100	150
NWA: *match* / *gap* / *mismatch*					
1/ 0/ −1	27	6*	5	5	5
1/ 0/ −1 (MA)	11	8	5	5	5
1/ 0/ −10	27	6*	5	5	5
1/ 0/ −10 (MA)	11	8	5	5	5
1/ 0/ 0	27	5	5	5	5
1/ 0/ 0 (MA)	11	7	5	5	5
1/ −4/ −6	27	5	5	5	5
1/ −4/ −6 (MA)	11	7	5	5	5
10/ 5/ 0	55	9	7	5	5
10/ 5/ 0 (MA)	34	13	9	5	5
1/ 0/ Eq. (2)	27	6*	5	5	5
1/ 0/ Eq. (2) (MA)	11	7	5	5	5
EDR	27	5	5	5	5

Table 4
NWA vs. EDR: classification results on Geolife dataset. * indicates correct classification. $\alpha = 0.75$ and $\beta = 3$. (MA) indicates results with Move ability.

	ϵ [m]				
	20	50	80	100	150
NWA: *match* / *gap* / *mismatch*					
1/ 0/ −1	81	8*	5	5	6
1/ 0/ −1 (MA)	26	9	5	5	6
1/ 0/ −10	81	8*	5	5	6
1/ 0/ −10 (MA)	26	9	5	5	6
1/ 0/ 0	60	7	5	5	6
1/ 0/ 0 (MA)	19	8*	5	5	6
1/ −4/ −6	60	7	5	5	6
1/ −4/ −6 (MA)	19	8*	5	5	6
10/ 5/ 0	130	8*	5	5	6
10/ 5/ 0 (MA)	42	10	5	5	6
1/ 0/ Eq. (2)	81	8*	5	5	6
1/ 0/ Eq. (2) (MA)	26	9	5	5	6
EDR	62	8*	5	5	5



(a) Two trajectories with linear interpolation.

(b) Two aligned trajectories, where green color indicates matches as computed by NWA.

Fig. 6. NWA: example.

number of applied mismatches is 0, whereas in the latter case mismatches get applied. Even in the case of equal classification accuracy, we would prefer the former option as evaluation of similarity does not depend on an additional variable.

Tables 6 and 7 show for a larger range of $(match, gap, mismatch)$ that, as formulated in Eq. (4), when applying NWA, mismatches are not necessary to obtain the optimum classification result for the two studied data sets.

9. Conclusion

We model match and mismatch score for NWA as a function of normal distribution and we show by means of experimental evaluation that optimal classification results can be achieved. Furthermore, we show that mismatches are not necessary in order to achieve optimum classification result. Instead, we show that only gaps (indels) are necessary. This basically leads to evaluating trajectories on basis of Longest Common Subsequence, even though, we apply some filtering in the form of low pass filter for consecutive gaps.

Not having to consider mismatches is a somewhat surprising result as it is often argued that longest common subsequence leads to inferior results (Chen et al., 2005). Our results indicate that longest common subsequence may lead to acceptable results, when gap length gets considered in evaluation.

Another surprising finding is that literature, wherein various string editing algorithms are applied, does not report whether gaps or mismatches are necessary for their particular problem; see for example (Chen et al., 2005; Zheng, 2015; Cleasby et al., 2019). Instead the results are reported in aggregated form, without any possibility to derive the number of applied gaps or mismatches.

Our conclusions are inline with Chen et al. (2005), wherein the authors state: *"Longest common subsequence can handle trajectories with noise, but it is a very "coarse" measure, as it does not differentiate trajectories with similar common subsequences but different sizes of gaps in between."* Our results show, however, that adding the possibility of mismatches (substitutions), which are for example applied in their EDR, does not alleviate the problem. Instead, it is necessary to further evaluate the results obtained with NWA without mismatches.

The challenges that we overcome, when applying NWA to user trajectory comparison, include challenges connected with evaluation of gaps (interrupted GPS service) and nests. We adopted the approach presented in Yang et al. (2020), where the authors assume that a trajectory consists of either move points, stay points or noise points (for example nests). NWA is an algorithm that is specifically designed for identifying similar parts of sequences, in our case its output is either a match or a gap. We herein assume that matches can be mapped to move points, leaving gaps for further analysis. We apply Move ability to analyze gaps, what could lead to an improved classification performance, as nests get effectively filtered out. We show however that our NWA approach enhanced with Move ability does not deliver significantly better results, even though, we observe some improvement for lower $\in$ values.

Table 5
NWA, Geolife: *#match*, *#gap* and *#mismatch* is the number of matches, the number of gaps and the number of mismatches, respectively.

match, gap, mismatch	$\in$ [m]	*#match*	*#gap*	*#mismatch*
1, 0, −1	2	137	1289	0
1, 0, −1	5	115	367	0
1, 0, −1	10	79	133	0
1, 0, −1	15	75	35	0
1, 0, −1	20	65	7	0
1, 0, −1	50	25	2	0
1, 0, −1	100	11	2	0
1, −1, 0	2	130	65	619
1, −1, 0	5	115	31	168
1, −1, 0	10	79	13	60
1, −1, 0	15	75	3	16
1, −1, 0	20	65	5	1
1, −1, 0	50	25	2	0
1, −1, 0	100	11	2	0

Table 6
NWA, COhave: *#match*, *#gap* and *#mismatch* is the number of matches, the number of gaps and the number of mismatches, respectively.

match, gap, mismatch	$\in$ [m]	*#match*	*#gap*	*#mismatch*
1, 0, −1	20	147990	614451	0
1, 0, −1	50	64318	214833	0
1, 0, −1	80	40068	123280	0
1, 0, −1	100	31968	94807	0
1, 0, −1	150	21074	55625	0
1, 0, −10	20	147990	614451	0
1, 0, −10	50	64318	214833	0
1, 0, −10	80	40068	123280	0
1, 0, −10	100	31968	94807	0
1, 0, −10	150	21074	55625	0
1, 0, 0	20	147990	196954	213349
1, 0, 0	50	64318	49992	84566
1, 0, 0	80	40068	34040	46550
1, 0, 0	100	31968	24220	37204
1, 0, 0	150	21074	11798	23673
1, −4, −6	20	146708	101600	262308
1, −4, −6	50	63826	38232	90938
1, −4, −6	80	39546	22116	53034
1, −4, −6	100	31546	16824	41324
1, −4, −6	150	21074	11788	23678
1, −1, 1	20	412462	94708	0
1, −1, 1	50	155398	36964	0
1, −1, 1	80	92678	21920	0
1, −1, 1	100	73174	16216	0
1, −1, 1	150	45294	10704	0
4, 3, −1	20	0	914871	0
4, 3, −1	50	0	342999	0
4, 3, −1	80	0	202515	0
4, 3, −1	100	0	157803	0
4, 3, −1	150	0	96531	0
−1, 0, 1	20	0	96027	411769
−1, 0, 1	50	0	39796	153945
−1, 0, 1	80	0	24924	91169
−1, 0, 1	100	0	19233	71613
−1, 0, 1	150	0	14275	43497
10, 5, 0	20	28455	858272	0
10, 5, 0	50	30867	281985	0
10, 5, 0	80	22020	159441	0
10, 5, 0	100	17912	122917	0
10, 5, 0	150	16109	65835	0
1, 0, Eq. (2)	20	147990	614451	0
1, 0, Eq. (2)	50	64318	214833	0
1, 0, Eq. (2)	80	40068	123280	0
1, 0, Eq. (2)	100	31968	94807	0
1, 0, Eq. (2)	150	21074	55625	0

Yet another challenge that we overcome is how to translate sequences of GPS positions to a letter-based representation that is an input to NWA. Missing GPS positions could lead to an outcome where NWA computes an identical alignment for two distinct trajectories. For this reason, we apply linear approximation of missing trajectory parts whenever possible.

The impact of our results can be summarized as follow. We apply NWA for user trajectory comparison, what has only received a limited interest from the research community. We analyze NWA efficiency when GPS user positions are normally distributed around user's actual position. This allows us to derive parameters that allow NWA to achieve optimal classification performance. We apply experimental evaluation based on two distinct data sets, COhave and Geolife, in order to provide support that NWA is suitable for user trajectory comparison. We explore the parameter space of our NWA approach, so that a more complex image about its performance is available. As NWA is a widely applied algorithm for global sequence alignment, it can be combined with other approaches, for example, TAD (Yang et al., 2020), which offer more

Table 7
NWA, Geolife: *#match*, *#gap* and *#mismatch* is the number of matches, the number of gaps and the number of mismatches, respectively.

match, *gap*, *mismatch*	∈ [m]	*#match*	*#gap*	*#mismatch*
1, 0, −1	20	3586755	16363957	0
1, 0, −1	50	1568876	5793302	0
1, 0, −1	80	995339	3412208	0
1, 0, −1	100	792775	2635524	0
1, 0, −1	150	538345	1597157	0
1, 0, −10	20	3586755	16363957	0
1, 0, −10	50	1568876	5793302	0
1, 0, −10	80	995339	3412208	0
1, 0, −10	100	792775	2635524	0
1, 0, −10	150	538345	1597157	0
1, 0, 0	20	3586755	5345516	5646667
1, 0, 0	50	1568876	1757690	2088799
1, 0, 0	80	995339	1057668	1236287
1, 0, 0	100	792775	846898	951756
1, 0, 0	150	538345	526118	594696
1, −4, −6	20	3565665	4439794	6120618
1, −4, −6	50	1566080	1681020	2129930
1, −4, −6	80	993511	1019626	1257136
1, −4, −6	100	792593	823118	963828
1, −4, −6	150	538333	517470	599032
1, −1, 1	20	9748879	4314602	0
1, −1, 1	50	3710240	1652560	0
1, −1, 1	80	2256617	1007686	0
1, −1, 1	100	1766365	803230	0
1, −1, 1	150	1142603	506994	0
4, 3, −1	20	0	23600760	0
4, 3, −1	50	0	8861440	0
4, 3, −1	80	0	5309320	0
4, 3, −1	100	0	4124360	0
4, 3, −1	150	0	2580600	0
−1, 0, 1	20	0	4343417	9733895
−1, 0, 1	50	0	1718804	3675630
−1, 0, 1	80	0	1118775	2198997
−1, 0, 1	100	0	907111	1711043
−1, 0, 1	150	0	679909	1054157
10, 5, 0	20	2255754	19108155	0
10, 5, 0	50	1351268	6223312	0
10, 5, 0	80	919207	3557581	0
10, 5, 0	100	716916	2777396	0
10, 5, 0	150	512906	1642963	0
1, 0, Eq. (2)	20	3586755	16363957	0
1, 0, Eq. (2)	50	1568876	5793302	0
1, 0, Eq. (2)	80	995339	3412208	0
1, 0, Eq. (2)	100	792775	2635524	0
1, 0, Eq. (2)	150	538345	1597157	0

insight on how dissimilar parts of sequences can be evaluated. We expect that when dissimilar parts of sequences get evaluated, user trajectory comparison can be further improved.

Our future research directions include a tight integration of our NWA approach with approaches that aim at identification of stay and noise points such as TAD (Yang et al., 2020). We hope, this could bridge our NWA approach with approaches that are more efficient in identifying areas where user subtrajectories show a high degree of dissimilarity. We will evaluate whether such an integration does not increase computational cost to a level that impedes our central goal – *efficient user trajectory comparison and clustering*.

Among our other future efforts, we would like to mention our ambition to analyze spatial activity of rodents that got tracked using RFID (Radio-frequency identification) technology; see Balogh, Bízik, Turčáni, and Koprda (2016) and Balogh and Baláž (2020). The rodents were chipped using PIT (passive integrated transponder) chips. Several RFID monitoring stations were deployed to detect rodent presence. By analyzing collected data we hope to be able to elucidate how rodents interact with their environment, possibly to provide similar insights on their social structure as reported by Mersch et al. (2013) in their investigation focused on ants.

CRediT authorship contribution statement

Maroš Čavojský: Conceptualization, Investigation, Methodology. **Martin Drozda:** Formal analysis, Supervision, Writing - original draft. **Zoltán Balogh:** Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

Martin Drozda was supported by the project "STU ako líder Digitálnej koalície", project No. 002STU-2–1/2018, financed by Ministry of Education, Science, Research and Sport of the Slovak Republic. Zoltán Balogh was supported in part by the project KEGA 036UKF-4/2019, Adaptation of the learning process using sensor networks and the Internet of Things.

References

1000 Genomes Project Consortium (2015). A global reference for human genetic variation. Nature 526, 68.
Ahn, H.-K., Knauer, C., Scherfenberg, M., Schlipf, L., & Vigneron, A. (2010). Computing the discrete Fréchet distance with imprecise input. In *International symposium on algorithms and computation* (pp. 422–433). Springer.
Balogh, Z., & Baláž, I. (2020). Optimizing of spatial activities monitoring using the Raspberry Pi and RFID system. In *Recent trends in intelligent computing, communication and devices* (pp. 615–622). Springer.
Balogh, Z., Bízik, R., Turčáni, M., & Koprda, Š. (2016). Proposal for spatial monitoring activities using the Raspberry Pi and LF RFID technology. In Wireless Communications, Networking and Applications (pp. 641–651). Springer.
Čavojský, M., & Drozda, M. (2016). Energy efficient trajectory recording of mobile devices using wifi scanning. In Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016 Intl IEEE Conferences (pp. 1079–1085).
Čavojský, M., & Drozda, M. (2019). Comparison of user trajectories with the Needleman-Wunsch algorithm. In *Proceedings of 10th EAI International Conference on Mobile Computing, Applications and Services (MOBICASE)* (pp. 1–13). Springer.
Čavojský, M., Uhlar, M., Ivanis, M., Molnar, M., & Drozda, M. (2018). User trajectory extraction based on wifi scanning. In *FiCloud 2018, The IEEE 6th International Conference on Future Internet of Things and Cloud* (pp. 115–120).
Chakraborty, A., & Bandyopadhyay, S. (2013). FOGSAA: Fast optimal global sequence alignment algorithm. *Scientific Reports, 3*, 1746.
Chen, L., Özsu, M.T., & Oria, V. (2005). Robust and fast similarity search for moving object trajectories. In Proceedings of the 2005 ACM SIGMOD international conference on Management of data (pp. 491–502). ACM.
Chua, S.-L., & Foo, L. K. (2017). Tree alignment based on Needleman-Wunsch algorithm for sensor selection in smart homes. *Sensors, 17*. URL: https://www.mdpi.com/1424-8220/17/8/1902.
Cleasby, I. R., Wakefield, E. D., Morrissey, B. J., Bodey, T. W., Votier, S. C., Bearhop, S., & Hamer, K. C. (2019). Using time-series similarity measures to compare animal movement trajectories in ecology. *Behavioral Ecology and Sociobiology, 73*, 151.
Day, R.-F. (2010). Examining the validity of the Needleman-Wunsch algorithm in identifying decision strategy with eye-movement data. *Decision Support Systems, 49*, 396–403.
Garhwal, A. S., & Yan, W. Q. (2019). BIIIA: A bioinformatics-inspired image identification approach. *Multimedia Tools and Applications, 78*, 9537–9552.
Google (2020). Location — Android Developers. URL: https://developer.android.com/reference/android/location/package-summary.html.
Güyer, T., Atasoy, B., & Somyürek, S. (2015). Measuring disorientation based on the Needleman-Wunsch algorithm. *International Review of Research in Open and Distributed Learning, 16*, 188–205.
Heng, L., Gao, G.X., Walter, T., & Enge, P. (2011). Statistical characterization of GPS signal-in-space errors. In Proceedings of the 2011 International Technical Meeting of the Institute of Navigation (ION ITM 2011), San Diego, CA (pp. 312–319). Citeseer.
Henikoff, S., & Henikoff, J. G. (1992). Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences, 89*, 10915–10919.
Hu, N., Dannenberg, R. B., & Tzanetakis, G. (2003). Polyphonic audio matching and alignment for music retrieval. In *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics* (pp. 185–188). IEEE.
Ju, S., Park, S., Lim, H., Yun, S.B., & Heo, J. (2018). Spatial-data-driven student characterization: Trajectory sequence alignment based on student smart card transactions. In Proceedings of the 2nd ACM SIGSPATIAL Workshop on Prediction of Human Mobility (pp. 1–7). ACM.

Luo, T., Zheng, X., Xu, G., Fu, K., & Ren, W. (2017). An improved DBSCAN algorithm to detect stops in individual trajectories. *ISPRS International Journal of Geo-Information, 6*, 63.

Mersch, D. P., Crespi, A., & Keller, L. (2013). Tracking individuals shows spatial fidelity is a key regulator of ant social organization. *Science, 340*, 1090–1093. URL: https://science.sciencemag.org/content/340/6136/1090.

Naidu, V., & Narayanan, A. (2016). Needleman-Wunsch and Smith-Waterman algorithms for identifying viral polymorphic malware variants. In *2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)* (pp. 326–333). IEEE.

Navarro, G. (2001). A guided tour to approximate string matching. *ACM Computing Surveys, 33*, 31–88. https://doi.org/10.1145/375360.375365

Needleman, S. B., & Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology, 48*, 443–453.

Ranacher, P., & Tzavella, K. (2014). How to compare movement? A review of physical movement similarity measures in geographic information science and beyond. *Cartography and Geographic Information Science, 41*, 286–307.

Rappaport, T. S. (1996). *Wireless communications: Principles and practice* (vol. 2). Prentice Hall PTR New Jersey.

Sellers, P. H. (1974). On the theory and computation of evolutionary distances. *SIAM Journal on Applied Mathematics, 26*, 787–793.

Tiberius, C., & Borre, K. (2000). Are GPS data normally distributed. In Geodesy Beyond 2000 (pp. 243–248). Springer.

Toohey, K., & Duckham, M. (2015). *Trajectory similarity measures. Sigspatial Special, 7*, 43–50.

Vineetha, V., Biji, C., & Nair, A. S. (2019). SPARK-MSNA: Efficient algorithm on Apache Spark for aligning multiple similar DNA/RNA sequences with supervised learning. *Scientific Reports, 9*, 6631.

Yang, D., Zhang, T., Li, J., & Lian, X. (2011). Synthetic fuzzy evaluation method of trajectory similarity in map-matching. *Journal of Intelligent Transportation Systems, 15*, 193–204.

Yang, Y., Cai, J., Yang, H., Zhang, J., & Zhao, X. (2020). Tad: A trajectory clustering algorithm based on spatial-temporal density analysis. *Expert Systems with Applications, 139*, Article 112846. https://doi.org/10.1016/j.eswa.2019.112846. URL: http://www.sciencedirect.com/science/article/pii/S0957417419305482.

Yuan, Y., & Raubal, M. (2012). Extracting dynamic urban mobility patterns from mobile phone data. In *International Conference on Geographic Information Science* (pp. 354–367). Springer.

Zheng, Y. (2015). Trajectory data mining: An overview. *ACM Transactions on Intelligent Systems and Technology (TIST), 6*, 29.

Zheng, Y., Fu, H., Xie, X., Ma, W. -Y., & Li, Q. (2011). Geolife GPS trajectory dataset - User Guide. URL: https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/.

Zheng, Y., Li, Q., Chen, Y., Xie, X., & Ma, W. -Y. (2008). Understanding mobility based on GPS data. In Proceedings of the 10th international conference on Ubiquitous computing (pp. 312–321). ACM.

Zheng, Y., Xie, X., & Ma, W.-Y. (2010). Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Engineering Bulletin, 33*, 32–39.

Zheng, Y., Zhang, L., Xie, X., & Ma, W. -Y. (2009). Mining interesting locations and travel sequences from gps trajectories. In Proceedings of the 18th international conference on World wide web (pp. 791–800). ACM.