Classification of particle trajectories in living cells: Machine learning versus statistical testing hypothesis for fractional anomalous diffusion

Joanna Janczura, Patrycja Kowalek, Hanna Loch-Olszewska, Janusz Szwabiński,[*] and Aleksander Weron
Faculty of Pure and Applied Mathematics, Hugo Steinhaus Center, Wrocław University of Science and Technology, 50-370 Wrocław, Poland

(Received 14 May 2020; revised 13 July 2020; accepted 27 July 2020; published 1 September 2020)

Single-particle tracking (SPT) has become a popular tool to study the intracellular transport of molecules in living cells. Inferring the character of their dynamics is important, because it determines the organization and functions of the cells. For this reason, one of the first steps in the analysis of SPT data is the identification of the diffusion type of the observed particles. The most popular method to identify the class of a trajectory is based on the mean-square displacement (MSD). However, due to its known limitations, several other approaches have been already proposed. With the recent advances in algorithms and the developments of modern hardware, the classification attempts rooted in machine learning (ML) are of particular interest. In this work, we adopt two ML ensemble algorithms, i.e., random forest and gradient boosting, to the problem of trajectory classification. We present a new set of features used to transform the raw trajectories data into input vectors required by the classifiers. The resulting models are then applied to real data for G protein-coupled receptors and G proteins. The classification results are compared to recent statistical methods going beyond MSD.

DOI: 10.1103/PhysRevE.102.032402

I. INTRODUCTION

Single-particle tracking (SPT) has become an important tool in the biophysical community in recent years. It was first carried out on proteins diffusing in the cell membrane [1,2]. Since then it was successfully used to study different transport processes in intracellular environment, providing valuable information about mechano-structural characteristics of living cells. For instance, it helped already to unveil the details of the movement of molecular motors inside cells [3,4] or of target search mechanisms of nuclear proteins [5].

Living cells belong to the class of active systems [6], in which the particles undergo simultaneous active and thermally driven transport. It has been shown already that the dynamics of proteins in cells determines their organization and functions [7]. This is the reason why it is crucial to identify the type of motion of the observed particles to deduct their driving forces [8–11].

Over the past decades, a number of stochastic models has been already proposed to describe the intracellular transport of molecules [11,12]. Within those models, the dynamics of molecules usually alternates between distinct types of diffusion, each of which may be associated with a different physical scenario. The Brownian motion [13] models a particle that diffuses freely, i.e., it does not meet any obstacles in its path nor it interacts with other molecules in its surrounding. The subdiffusion is appropriate to represent trapped particles [11,14], particles which encounter fixed or moving obstacles [8,15] or particles slowed down due to the viscoelastic properties of the cytoplasm [16]. Finally, the superdiffusion models the motion driven by molecular motors: the particles move faster than in a free diffusion case and in a specific direction [17]. The sub- and superdiffusion together are often referred to as the anomalous diffusion.

The standard method of classification of individual trajectories into those three types of diffusion is based on the mean-square displacement (MSD) [12]. Within this approach one fits the theoretical MSD curves for various models to the data and then selects the best fit with statistical analysis [18]. A linear MSD curve indicates the free diffusion, a sublinear (superlinear) one - the subdiffusion (the superdiffusion). However, there are some issues related with this method. In many cases, the experimental trajectories are too short to extract a meaningful information from MSD. Moreover, the finite precision adds a term to the MSD, which is known to limit the interpretation of the data [9,12,19,20]. As a result, several methods improving or going beyond the MSD have been introduced to overcome these problems. For instance, Michalet [12] used an iterative method called the optimal least-squares fit to determine the optimal number of points to obtain the best fit to MSD in the presence of localization errors. Weiss [21] used a resampling approach that eliminates localization errors in the time-averaged MSD of subdiffusive fractional Brownian motion processes. The trajectory spread in space calculated through the radius of gyration [22], the Van Hove displacements distributions and deviations from Gaussian statistics [23], self-similarity of trajectory using different powers of the displacement [24], velocity autocorrelation function [25,26] or the time-dependent directional persistence of trajectories [27] methods can be combined with the output of MSD to improve the classification results. The distribution of directional changes [28], the mean maximum excursion method [29] and the fractionally integrated moving average (FIMA) framework [30] may efficiently replace the MSD estimator for classification purposes. Hidden Markov Models (HMM) has been proposed to check the heterogeneity within single trajectories [31,32]. They have proven to be

[*]Corresponding author: janusz.szwabinski@pwr.edu.pl

quite useful in the detection of confinement [33]. Last but not least, classification based on hypothesis testing, both relying on MSD and going beyond this statistics, has been shown to be quite successful as well [20,34].

An alternative, very promising approach to SPT data analysis is rooted in computer science. Namely, classification of trajectories may be seen as a subject of machine learning (ML) [35]. In the ML context, classification relies on available data, because its goal is to identify to which category a new observation belongs on the basis of a training data set containing observations with a known category membership.

There is already a number of attempts to analyze particle trajectories with machine learning methods. Among them, Bayesian approach [18,36,37], random forests [38–40], neural networks [41], and deep neural networks [39,42–44] have gained a lot of attention and popularity. While some of the works have focused just on the identification of the diffusion modes [38,39,41], the others went beyond just the classification of diffusion and tried to extract quantitative information about the trajectories (e.g., the anomalous exponent [40,42]).

Recently, we presented a comparison of performance of two different classes of methods: traditional feature-based algorithms (random forest and gradient boosting) and a modern deep learning approach based on convolutional neural networks [39]. The latter constitutes nowadays the state-of-the-art technology for automatic data classification and is much simpler to use from the perspective of the end-user, because it operates on raw data and does not require any preprocessing effort from human experts [45]. In contrast, the traditional methods require a representation of trajectories by a set of human-engineered features or attributes [46]. In most of the applications the deep learning approach outperforms the traditional methods. However, in some situations it is still worth to use them, because they usually work better on small data sets, are computationally cheaper and easier to interpret. From our results it follows that both approaches achieve excellent (and very similar) accuracies on synthetic data. But they turned out to be really bad in terms of transfer learning. This concept refers to a situation, in which a classifier is trained in one setting and then applied to a different one. The classifiers from Ref. [39] were not able to successfully classify trajectories generated with methods different from the ones used for the training set.

In this paper, we are going to present an improved version of the traditional classifiers presented in Ref. [39]. We will propose a new set of training data as well as a new collection of features describing a trajectory. Both are inspired by a recent statistical analysis of anomalous diffusion [34]. To illustrate the transfer learning abilities of the new classifiers, we will apply them to the data from a single-particle tracking experiment on G protein-coupled receptors and G proteins [47]. Results of classification from Ref. [34] will be used as a benchmark.

The paper is organized as follows. In Sec. II, we briefly introduce the different modes of diffusion and methods of their analysis. Section III contains a short description of the machine learning methods used in this work. Stochastic models of diffusion for generation of synthetic data are presented in Sec. IV. The data itself is characterized in Sec. V. The set of features used as input to the classifiers is introduced in Sec. VI. Our results are presented in Sec. VII, followed by some concluding remarks.

II. DIFFUSION MODES AND THEIR ANALYSIS

As already mentioned in the Introduction, identification of the diffusion modes of particles within living cells is important, because they reflect the interactions of those particles with their surrounding. For instance, if a particle is driven by a free diffusion (Brownian motion) [13], then we expect that it does not meet any obstacles in its path and does not undergo any relevant interactions with other particles. Deviations from Brownian motion are called anomalous diffusion and can be divided into two distinct classes. Subdiffusion is slower than the normal one. It usually occurs in crowded or constrained domains and can be brought together with different physical mechanisms including immobile obstacles, cytoplasm viscosity, crowding, trapping, and heterogeneities [48–50]. Superdiffusion represents active transport along the cytoskeleton, assisted by molecular motors [17]. Particles undergoing that type of motion move faster than those freely diffusing and usually do not come back to previous positions.

Although different scenarios for both classes of anomalous diffusion are possible [11,49,51–54], for the purpose of this work we will limit ourselves to those three basic types mentioned above: free, sub-, and superdiffusion.

The most popular method of deducing a particles' type of motion from their trajectories is based on the analysis of the MSD [55],

$$\rho(t) = E(\|X_{t+t_0} - X_{t_0}\|^2), \tag{1}$$

where $(X_t)_{t>0}$ is a particle trajectory, $\|\cdot\|$ is the Euclidean norm, and E is the expectation of the probability space. Since in many experiments only a limited number of trajectories is observed, the time-averaged MSD (TAMSD) calculated from a single trajectory is usually used as the estimator of MSD,

$$\widehat{\rho}(n\Delta t) = \frac{1}{N-n+1}\sum_{i=0}^{N-n} \left\|X_{t_{i+n}} - X_{t_i}\right\|^2. \tag{2}$$

The trajectory is assumed to be given in the form of N consecutive two-dimensional positions $X_i = (x_i, y_i)$ $(i = 0, \ldots, N)$ recorded with a constant time interval Δt and n is the time lag between the initial and the final positions of the particle. If the underlying process is ergodic and has stationary increments, then TAMSD converges to the theoretical MSD [51].

TAMSD as a function of the time lag for the normal diffusion converges asymptotically to a linear function [9], i.e., for large N:

$$\widehat{\rho}(n\Delta t) \sim 4D(n\Delta t), \tag{3}$$

with D being the diffusion coefficient. For subdiffusion, being slower than diffusion, the behavior of TAMSD is sublinear, while for superdiffusion, being faster than diffusion, the behavior is superlinear. Thus, for pure trajectories with no localization errors, one could easily determine their diffusion type by fitting a function $\alpha \ln(n\Delta t) + \beta$ to the estimated $\ln[\widehat{\rho}(n\Delta t)]$ curve. If $\alpha < 1$, then the trajectory can be identified as subdiffusive, while if $\alpha > 1$, then the trajectory can be identified as superdiffusive. Although theoretically this approach allows for the uncomplicated distinction of the diffusion types, there are several issues related with it as a method for classification. First, real trajectories are usually noisy, which makes the fitting of a mathematical model a challenging task, even in the simplest case of the normal diffusion [12,21]. Second, according to Eq. (2), only the values of $\widehat{\rho}$ corresponding to small time lags are well averaged. The larger the lag, the smaller is the number of displacements contributing to the averages, resulting in fluctuations increasing with the lag. Selecting a suitable lag is by the way a well known problem in biophysics [20,56,57]. Since many real trajectories are short, we are forced to concentrated on short times (small lags). This induces another problem in a classification method based only on MSD curves, as in this case the different power laws look alike even in the absence of noise.

III. MACHINE LEARNING APPROACH

Several different procedures have been already proposed to circumvent the limitations of the MSD [12,20,22–24,27–29,31–34], including the use of machine learning methods [18,36–40,42,43]. Recently, we discussed the applicability of three different machine learning algorithms to classification, including two feature-based methods and a deep learning one [39]. The results of that study were ambiguous. On the one hand, all of the methods performed excellent on the test data; on the other hand, they failed to transfer their knowledge to data coming from unseen physical models. The latter finding practically disqualified them as candidates for a reliable classification tool.

In this paper, we are going to continue the analysis started in Ref. [39] and present improved versions of the classifiers, which performs much better in terms of transfer learning. We will focus on the traditional machine learning methods: the random forest (RF) [58,59] and the gradient boosting (GB) [60,61]. Both methods are feature-based, meaning that each instance in the data set is described by a set of human-engineered attributes [46]. And both belong to the class of ensemble methods, which combine multiple base classifiers to form a better one. In each case, decision trees [62] are used as the base classifiers.

A decision tree is built by splitting the original dataset (trajectories with known classes), constituting the root node of the tree, into subsets, which represent the successor children. The splitting is based on a set of rules utilizing the values of features. This process is repeated on each derived subset in a recursive manner. The recursion is completed when the subset at a node has all samples belonging to the same class (i.e., the node is pure) or when splitting no longer adds value to the classification. At each step, a feature that best splits the data is chosen. Two metrics are typically used to measure the quality of the split: Gini impurity and information gain [35].

Gini impurity tells us how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in that set. It is given by

$$I_G = \sum_{i=1}^{J} p_i(1 - p_i), \tag{4}$$

where J is the number of classes ($J = 3$ in our case) and p_i is the fraction of items labeled with class i in the set.

Information gain related to a split is simply the reduction of information entropy [63], calculated as the difference between the entropy of a parent node in the tree and a weighted sum of entropies of its children nodes. The entropy itself is given as

$$H = -\sum_{i=1}^{J} p_i \log_2 p_i, \tag{5}$$

where $p_1, p_2, \ldots$ are fractions of each class present in the node.

Decision trees are often used for classification purposes, because they are easy to understand and interpret. However, single trees are unstable in the sense that a small variation in the data may lead to a completely different tree [64]. They also have a tendency to overfit, i.e., they model the training data too well and learn noise or random fluctuations as meaningful concepts, which limits their accuracy in case of unseen data [65]. That is why they are rather used as building blocks of the ensembles and not as stand alone classifiers.

In a random forest, multiple decision trees are constructed independently from the same training data. The predictions of individual trees are aggregated and then their mode is taken as the final output. In gradient boosting, the trees are not independent. Instead, they are built sequentially by learning from mistakes commited by the ensemble. In many applications, gradient boosting is expected to have a better performance than random forest. However, it is usually not the better choice in case of very noisy data.

A workflow of our classification method is shown in Fig. 1. The training set consists of a large number of synthetic trajectories and their labels (diffusion modes). The trajectories were generated with various kinds of theoretical models of diffusion (see Secs. IV and V B for further details). In the preprocessing phase, the raw data is cleaned and transformed into a form required as input by the classifier. Many traditional classifiers including random forest and gradient boosting work much better with vectors of features characterizing each trajectory instead of raw data. The features used in this work are introduced in Sec. VI. Some authors normalize the trajectories before further processing [40]. However, we omitted this step as our preliminary analysis indicated a significant decrease in the performance of the classifiers induced by normalization. The ensembles of trees were inferred from the feature vectors and their labels. Once trained, they may be used to classify new trajectories, including the experimental ones.

IV. STOCHASTIC MODELS OF DIFFUSION

The most popular theoretical models of diffusion commonly employed are: continuous-time random walk (CTRW) [11], obstructed diffusion (OD) [8,66], random walk on random walks (RWRW) [67], random walks on percolating clusters (RWPC) [68,69], fractional Brownian motion (FBM) [70–72], fractional Levy α-stable motion (FLSM) [73], fractional Langevin equation (FLE) [74], and autoregressive fractionally integrated moving average (ARFIMA) [75]. They are applicable to different physical environments: trapping and crowded environments (CTRW,

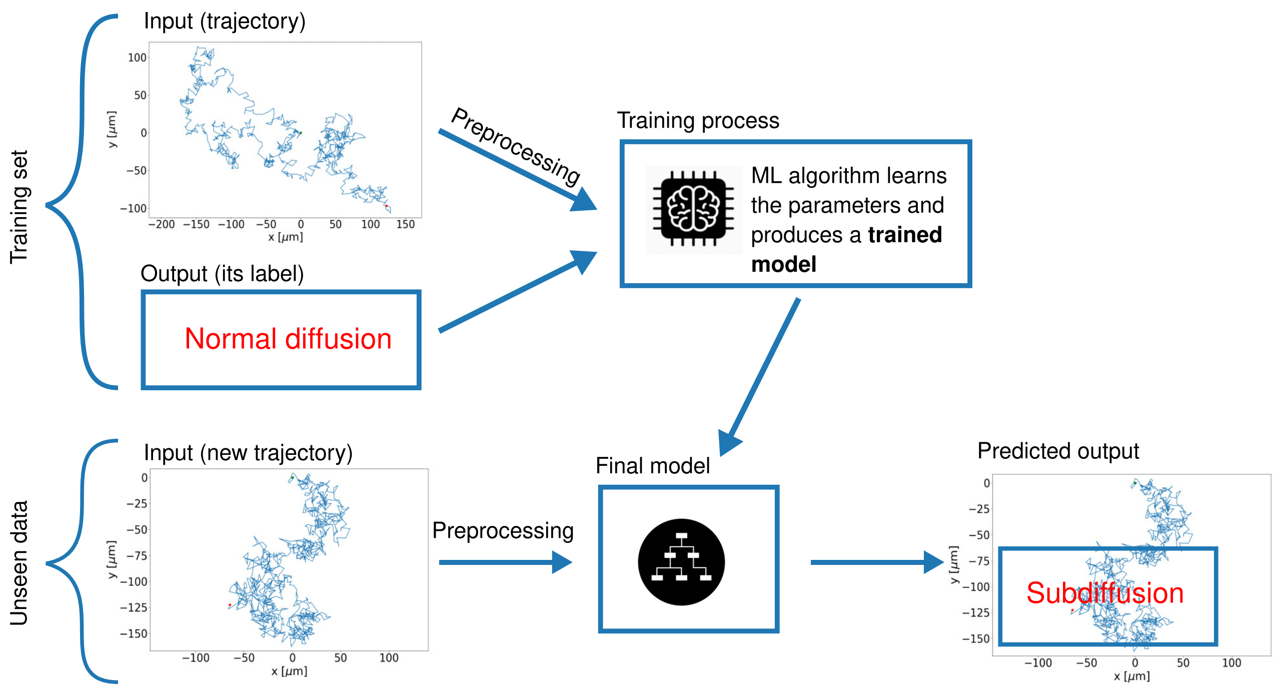

FIG. 1. Workflow of our classification method. The training set is composed of a large number of synthetic trajectories (Sec. V B). The preprocessing phase consists in extraction of features introduced in Sec. VI.

FFPE); labyrinthine environments (OD, RWPC, RWRW); viscoelastic systems (FBM, FLSM, FLE, ARFIMA); systems with time-dependent diffusion (scaled FBM, ARFIMA). Following Refs. [20,34], we will focus on three stochastic processes known to generate different kinds of fractional diffusion: fractional Brownian motion, directed Brownian motion (DBM) [76], and Ornstein-Uhlenbeck process (OU) [77].

FBM is the solution of the stochastic differential equation

$$dX_t^i = \sigma dB_t^{H,i}, \quad i = 1, 2, \tag{6}$$

where the parameter $\sigma > 0$ relates to the diffusion coefficient via $\sigma = \sqrt{2D}$, H is the Hurst parameter ($H = \alpha/2$), and B_t^H—a continuous-time Gaussian process that starts at zero, has expectation zero and has the following covariance function:

$$E\big(B_t^H B_s^H\big) = \tfrac{1}{2}\big(|t|^{2H} + |s|^{2H} - |t - s|^{2H}\big). \tag{7}$$

For $H < \frac{1}{2}$ (i.e., $\alpha < 1$), FBM produces subdiffusive trajectories. It corresponds to a scenario, in which a particle is hindered by mobile or immobile obstacles [78]. It reduces to the free diffusion at $H = \frac{1}{2}$. And for $H > \frac{1}{2}$, FBM generates superdiffusive motion [Fig. 2(a)].

The directed Brownian motion, also known as the diffusion with drift, is the solution to

$$dX_t^i = v_i dt + \sigma dB_t^{1/2,i}, \quad i = 1, 2, \tag{8}$$

where $v = (v_1, v_2) \in \mathbf{R}^2$ is the drift parameter. This process generates superdiffusion related to an active transport of particles driven by molecular motors. The velocity of the motors is modeled by the parameter v [Fig. 2(b)]. For $v = 0$, the process reduces to normal diffusion.

The Ornstein-Uhlenbeck process is known to model confined diffusion, which is a subclass of subdiffusion [Fig. 2(b)]. It corresponds to a particle inside a potential well and is a solution to the following stochastic differential equation:

$$dX_t^i = -\lambda_i\big(X_t^i - \theta_i\big)dt + \sigma dB_t^{1/2,i}, \quad i = 1, 2, \quad \theta_i \in \mathbf{R}. \tag{9}$$

Here, $\theta = (\theta_1, \theta_2)$ is the equilibrium position of the particle and λ_i measures the strength of interaction. For $\lambda_i = 0$, OU reduces to normal diffusion as well.

V. OUR DATASET

A. Real SPT data

The classifiers built in this study will be applied to the data from single-particle tracking experiment on G protein-coupled receptors and G proteins, already analyzed in Refs. [34,47]. The receptors are of great interest, because they mediate the biological effects of many hormones and neurotransmitters and are also important as pharmacological targets [79]. Their signals are transmitted to the cell interior via interactions with G proteins. The analysis of the dynamics of these two types of molecules will shed more light on how the receptors and G proteins meet, interact and couple.

A subset of that data has been already studied by means of statistical methods in Ref. [34]. Since we are interested in using those results as a benchmark for our classifiers, we will focus on the very same subset of data in our analysis. Hence, only trajectories with at least 50 steps will be taken into account, resulting in 1037 G proteins and 1218 receptors. The trajectories under consideration for both types of molecules are visualized in Fig. 3.

B. Synthetic data

Building a classifier requires training data, which consists of a set of training examples [35]. Each of these examples is

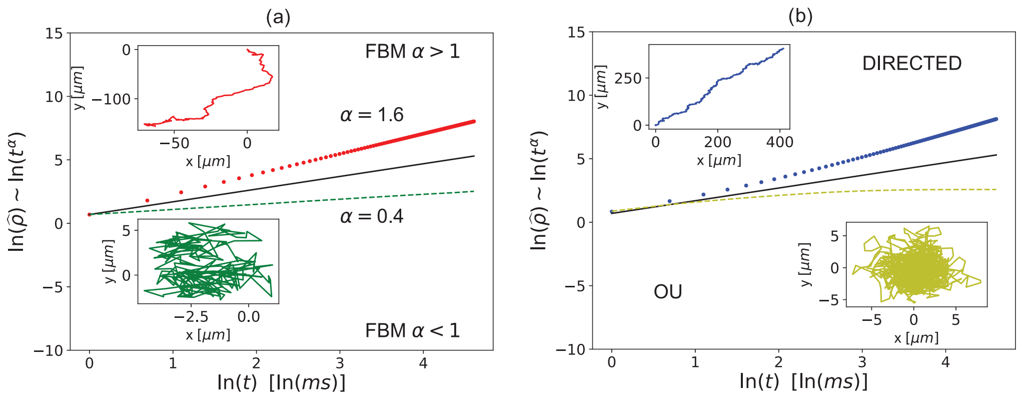

FIG. 2. Time-averaged mean-square displacement calculated for: (a) FBM with different values of α, (b) DBM and OU processes. The trajectories used to calculate the MSD curves are shown in the corresponding insets and are consistent with real data time and distance scales: ms and μm accordingly. The solid line in both plots indicates TAMSD of normal diffusion.

a pair of an input (trajectory) and its output label (diffusion type). In an optimal scenario the training set would contain real trajectories with their true labels from, e.g., previous experiments on the same type of cells. However, collecting a training set consisting of real trajectories is practically impossible. First of all, independently of the method used for analysis, the labels of such trajectories are affected by some uncertainties [34]. Moreover, typical machine learning algorithms require thousands of training examples to provide a reasonable function that maps an input to an output and can be used for classification of new input data. That is why one usually resorts to synthetic, computer generated trajectories to prepare the training set. In this case the true label of each trajectory is known in advance and it is rather cheap to generate many of them.

The stochastic processes described in Sec. IV will be used to generate the training set. Just to recall, a discrete trajectory of a particle is given by

$$X_n = \left(X_{t_0}, X_{t_1}, \dots, X_{t_N}\right), \tag{10}$$

where $X_{t_i} = (X_{t_i}^1, X_{t_i}^2) \in \mathbf{R}^2$ is the position of the particle at time $t_i = t_0 + i\Delta t$, $i = 0, 1, \dots, N$. The lag Δt between two consecutive observations is assumed to be constant. In tracking experiments, it is determined by the temporal resolution of the imaging method. However, we will assume the lag being equal to $1\,s$ in the simulations. Similarly, we will use $\sigma = 1\,\mu\mathrm{m\,s}^{-1/2}$ most of the time (see Sec. V D for an exception to this choice). In total, 120 000 trajectories have been generated for the main training set. Their length was randomly chosen from the range between 50 and 500 steps. No additional noise was added to the raw data in this set (see Sec. V C for a set with noise).

A summary of the training set is presented in Table I. The case of the free diffusion requires probably a short explanation. From the description in Sec. IV we know that all of the models reduce to the normal Brownian motion for some specific values of the parameters ($H = 0.5$ for FBM, $v = 0$ for DBM, and $\lambda = 0$ for OU). However, it is very difficult to distinguish anomalous diffusion processes from the normal one already if their parameters are in the vicinity of those values. That is why we extended the ranges of parameter values

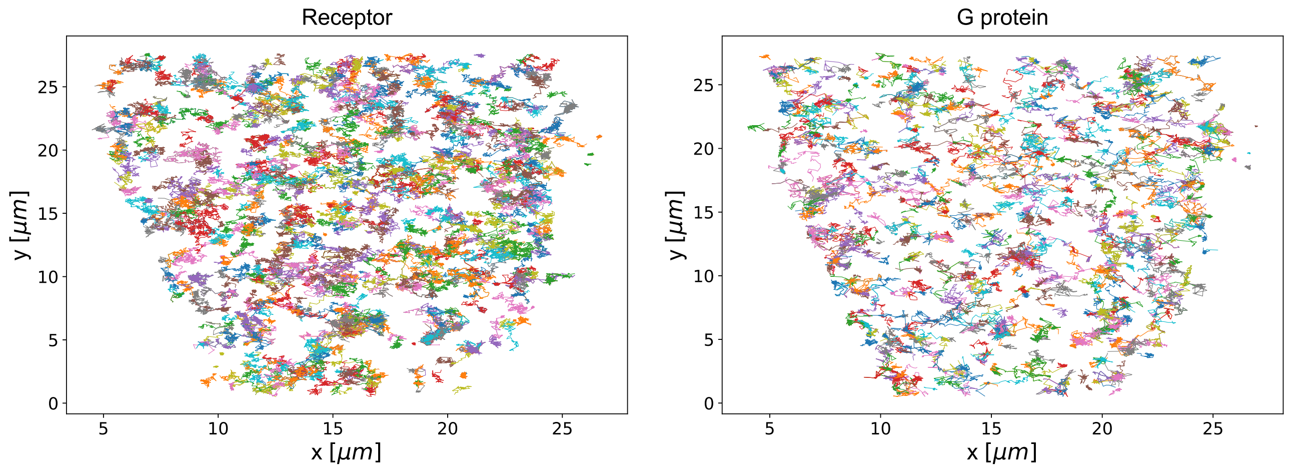

FIG. 3. Trajectories of the receptors (left) and G proteins (right) used as input for the classifiers. Different colors are introduced to indicate different trajectories. The set of the receptors contains 1218 trajectories and the one of G proteins—1037 trajectories. The lengths of the trajectories are from range [50, 401], the time step is equal to 28.4 ms, and recorded positions are given in μm.

TABLE I. Summary of the synthetic trajectories used as the training set. The parameter c was set to 0.1 in all simulations. If not specified otherwise, then $\sigma = 1\,\mu\mathrm{m\,s}^{-1/2}$ and $\Delta t = 1$ s were used.

Type of diffusion	Model	Parameter ranges	Number of trajectories
Normal diffusion	FBM	$H \in [0.5 - c, 0.5 + c]$	20 000
	DBM	$v \in [0, c]$	10 000
	OU	$\theta = 0, \lambda = [0, c]$	10 000
Subdiffusion	FBM	$H \in [0.1, 0.5 - c)$	20 000
	OU	$\theta = 0, \lambda = (c, 1]$	20 000
Superdiffusion	FBM	$H \in (0.5 + c, 0.9]$	20 000
	DBM	$v \in (c, 1]$	20 000

corresponding to the free diffusion. Although introduced here at a different level, this approach resembles the cutoff c used in Ref. [34] to classify the results. As for the value, we take the smallest one considered therein.

The Python package `fbm` [80] was used to simulate the FBM trajectories as well as the Brownian motion part of the diffusion with drift. By default, the `fbm()` function from that package utilizes the Davies-Harte method [81] for fastest performance. However, the method is known to fail for the Hurst parameter close to 1. If this occurs, then the function fallbacks to the Hosking's algorithm [82]. The OU process was generated with the `OrnsteinUhlenbeckProcess` object from the `stochastic` package [83]. This object uses the Euler-Maruyama method [84] to produce realizations of the process.

C. Adding noise

The synthetic dataset introduced in the previous section constitutes our main training set for building the classifiers. For the sake of comparison with statistical methods presented in Ref. [34], it consists of pure trajectories, that do not suffer from any localization errors.

However, real data is usually affected by different kinds of noise. For instance, slow drift currents in the cytoplasm may induce low frequency noise. Typically, it may be reduced by various detrending methods [12,85]. In contrast, high-frequency noise can be due to a variety of reasons: mechanical vibrations of the instrumental setup; particle displacement while the camera shutter is open; noisy estimation of true position from the pixelated microscopy image; error-prone tracking of particle positions when they are out of the camera focal plane [21,30,86–88].

To account for different localization errors and to check their impact on the performance of the classifiers, we prepared a second training set by simply adding a normal Gaussian noise with zero mean and standard deviation σ_{gn} to each simulated trajectory. We followed the procedure already used in Refs. [38,39]. That means, instead of setting σ_{gn} directly, we first introduced the signal to noise ratio,

$$Q = \begin{cases} \frac{\sqrt{D\Delta t + v^2\Delta t^2}}{\sigma_{\mathrm{gn}}} & \text{for DBM,} \\ \frac{\sqrt{D\Delta t}}{\sigma_{\mathrm{gn}}} & \text{otherwise,} \end{cases} \tag{11}$$

where $v = \sqrt{v_1^2 + v_2^2}$. For each trajectory, Q was randomly set in the range from 1 (high noise) to 9 (low one). Then, Eq. (11) was used to determine σ_{gn} for given D and Δt.

D. Auxiliary training set

During our first attempt to apply the classifiers to experimental data it turned out that the parameter $\sigma = 1$, taken from Ref. [34], may not be the best choice for real trajectories under investigation. Thus, we also prepared an auxiliary training set of synthetic trajectories, which were simulated with no noise and $\sigma = 0.38$. This particular value of σ corresponds to the diffusion coefficient $D = 0.0715\,\mu\mathrm{m}^2\,\mathrm{s}^{-1}$ and will be explained in Sec. VII E 3. All other parameters of the set are exactly the same as in the main training set introduced in Sec. V B.

VI. CLASSIFICATION FEATURES

The random forest and gradient boosting algorithms require human-engineered features representing the trajectories for both the model training and the classification of new data. Choosing the right features constitutes a challenge and is crucial for the classification results. For instance, in Ref. [39] we considered a set of features proposed for the first time by Wagner *et al.* [38]. Although we did not apply them to real data, we showed that classifiers using those features do not generalize well to data generated with models different from the ones used for training.

A more detailed discussion on the role of the features will be addressed in a forthcoming paper. In this work we use a new set of features motivated by the statistical analysis carried out in Ref. [34]. The main conclusion of that paper was that, even though statistical methods going beyond the standard MSD classification may provide good results even for short trajectories, no method was found to be superior in all examples and one should actually combine different approaches to get reliable results.

Following this recommendation, we decided to extract features from all methods considered in Ref. [34] and to use them simultaneously as the input for our classifiers. Thus, our feature set will consist of:

(1) anomalous α exponent (fitted to TAMSD),
(2) the diffusion coefficient D (fitted to TAMSD),

(3) the standarized value

$$T_N = \frac{D_N}{\sqrt{\hat{\sigma}_N^2 (t_N - t_0)}} \tag{12}$$

of the maximum distance D_N traveled by a particle,

$$D_N = \max_{i=1,2,\dots,N} \|X_{t_i} - X_{t_0}\|, \tag{13}$$

where $\hat{\sigma}_N$ is a consistent estimator of the standard deviation of D_N,

$$\hat{\sigma}_N^2 = \frac{1}{2N\Delta t} \sum_{j=1}^{N} \|X_{t_j} - X_{t_{j-1}}\|_2^2, \tag{14}$$

(4) the power γ^p (in the function kn^{γ^p}) fitted to p variation [73,89]

$$\hat{V}_n^{(p)} = \sum_{k=0}^{N/n-1} \|X_{(k+1)n} - X_{kn}\|^p, \tag{15}$$

for values of p from 1 to 5.

Note that the first two of the above features were included in the feature set used in Refs. [39]. To determine their values, the maximum lag equal to 10% of each trajectory's length was used to calculate the corresponding TAMSD curve.

VII. RESULTS

We used the `scikit-learn` [90] implementations of the random forest and gradient boosting algorithms. As already stated in Ref. [39], a cluster of 24 CPUs with 25 GB total memory was used to perform the computation. The processing time (feature extraction, hyperparameter tuning, training and validation of a model) was of the order of 2 h in each case. If not stated otherwise, then the dataset without noise (see Sec. V B) was used to train the classifiers.

A. Details of the classifiers

To find optimal models, we used the `RandomisedSearchCV` method from `scikit-learn` library. It allows us to perform a search over a grid of hyperparameter ranges. Here, a hyperparameter of the model is understood as a parameter, whose value is set before the learning process begins (it cannot be derived simply by training of the model).

In Table II, the optimal values of the hyperparameters for our training set are listed. The "with D" column in the table refers to the full set of features defined in Sec. VI. The "no D" columns corresponds to a reduced feature set with the diffusion coefficient D removed from consideration. The reason for introducing the latter set will be explained in Sec. VII E. The `bootstrap` hyperparameter is a boolean value. It decides whether bootstrap samples (True) or the whole data set (False) are used to build each single tree. `Criterion` specifies, which function should be used to measure the quality of a split of data into subsamples at a new node of the tree. Gini impurity and information entropy are available for that purpose [35]. The `max_depth` is the maximum depth (the number of levels) of each decision tree. The number of features to consider when looking for the best split is given by `max_features`. If equal to `log2` (`sqrt`), then the number is calculated as the logarithm (square root) of the number of features. `Min_samples_split` specifies the minimum number of samples required to split a subset of data at an internal node of the tree. `Min_samples_leaf` is the minimum number of samples required to be at a leaf node (a node representing a class label). Finally, `n_estimators` gives the number of trees in the ensemble.

TABLE II. Hyperparameters of the optimal classifiers found with both methods. Their meaning is explained in Sec. VII A. The "With D" column refers to the full feature set, "No D" one—to the feature set after removal of the diffusion coefficient D. NA stands for "Not applicable" (the first two parameters are random forest specific).

	Random forest		Gradient boosting	
	With D	No D	With D	No D
`bootstrap`	True	True	NA	NA
`criterion`	Entropy	Entropy	NA	NA
`max_depth`	60	10	10	10
`max_features`	$\log_2$	sqrt	$\log_2$	$\log_2$
`min_samples_leaf`	4	2	2	2
`min_samples_split`	2	10	2	2
`n_estimators`	900	600	100	100

As it follows from Table II, the ensemble found with gradient boosting is significantly smaller than the one generated with the random forest method.

B. Performance of the classifiers

Since our synthetic data set is perfectly balanced (same number of trajectories in each class), we may start the analysis of the classifiers simply by looking at their accuracy. It is one of the basic measures to assess the classification performance, defined as the number of correct predictions divided by the total number of preditions.

From the results listed in Table III it follows that in the case of the training set, the gradient boosting method is the more accurate one, even though the differences are small. Moreover, its decline in accuracy after the removal of D from the feature set is smaller than for random forest. However, the latter one performs a little bit better on the test set, indicating a small tendency of GB to overfit. Despite these differences, both classifiers perform very well.

The normalized confusion matrices of the classifiers are presented in Fig. 4. By definition, an element C_{ij} of the confusion matrix is equal to the number of observations known to be in class i (true labels) and predicted to be in class j (predicted labels) [35]. In all cases, the worst performance (93% of correctly predicted labels) is observed for the normal

TABLE III. Accuracies of the optimal classifiers for both the training and the test data.

	Random forest		Gradient boosting	
Data set	With D	No D	With D	No D
Training	0.977	0.953	0.992	0.989
Test	0.948	0.946	0.947	0.944

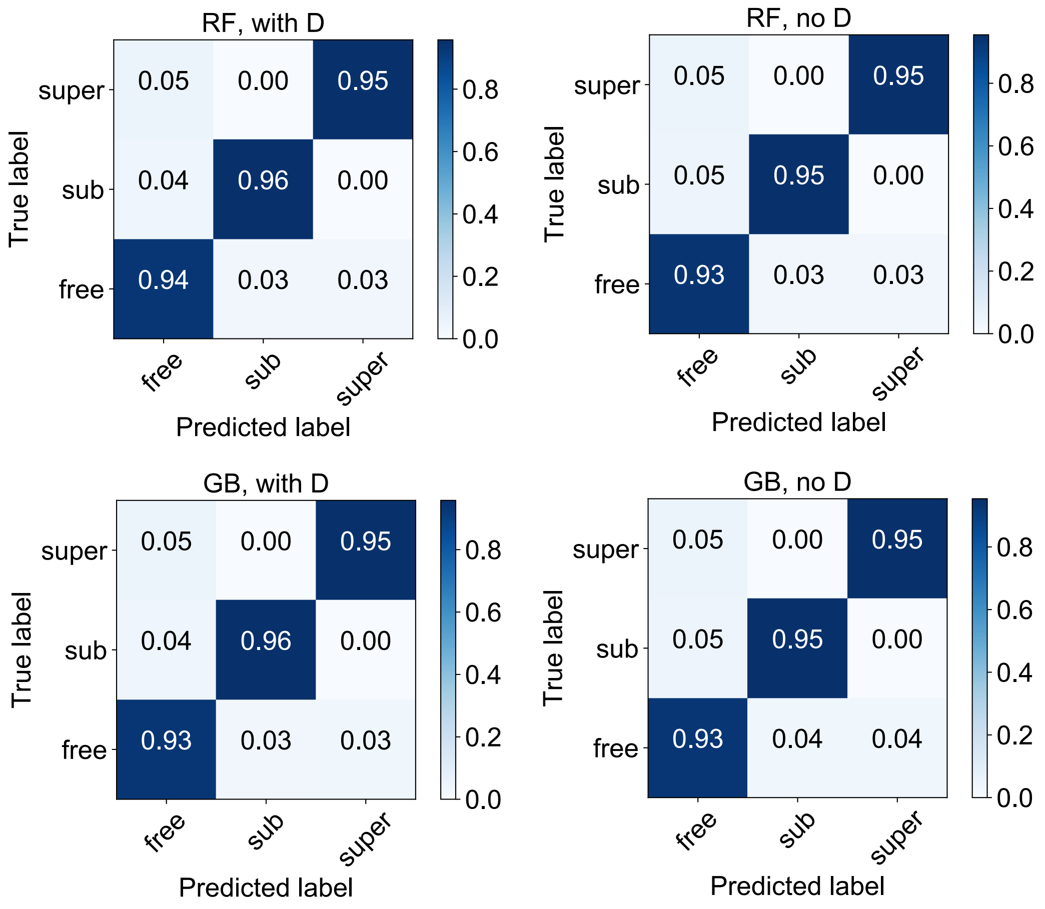

FIG. 4. Normalized confusion matrices for random forest and gradient boosting classifiers. The "With D" label refers to the full feature set, "No D" one—to the feature set after removal of the diffusion coefficient D. All results are rounded to two decimal digits.

diffusion. This relates probably to the fact that in our synthetic training set we also tagged anomalous trajectories with parameters slightly deviating from the normal ones as free diffusion.

The data collected in Fig. 4 may be used to calculate some other popular measures giving more insight into the performance of the classifiers: precision, recall, and F1 score [91]. Precision is the fraction of correct predictions of a class among all predictions of that class. It tells us how often a classifier is correct if it predicts a given class. Recall is the fraction of correct predictions of a given class over the total number of members of that class. It measures the number of relevant results within a predicted class. A harmonic mean of precision and recall gives the F1 score—another measure of classifier's accuracy.

As we see from Table IV, both models return much more relevant results than the irrelevant ones (high precision). Moreover, they yield most of the relevant results (high recall). The F1 scores resemble the accuracies given in Table III.

C. Feature importances

While working with the human-engineered features, it may happen that some of them are more informative than the others. Therefore, knowing the relative importances of the features is useful, because it can provide further insight into the data and the classification model. The features with high importances are the drivers of the outcome. The least important ones might often be omitted from the model, making it faster to fit and predict. The latter is of particular significance in case of models with very large feature sets, as it may additionally help to reduce the dimensionality of the problem.

There are several ways to determine the feature importances. The one implemented in the `scikit-learn` library is defined as the total decrease in node impurity caused by a given feature, averaged over all trees in the ensemble [92]. In other words, the Gini impurities Eq. (4) are calculated before and after each split on a given feature to determine the total decrease in the impurity related to that feature. The outcome is then averaged over all trees in the ensemble.

Relative feature importances for both classifiers are shown in Table V. The features are ordered according to the descending scores in case of the random forest with D.

We see that the p-variation for $p = 2$ (2-var) is the most informative feature, followed by the anomalous exponent α. The diffusion coefficient is the least important feature. After its removal the relative importances of the remaining features changed. The differences between them are smaller now. Moreover, the 1-var became the second most important attribute and T_N—the least important one.

Gradient boosting differs slightly from the random forest. In case with D, the order of the top two features is reversed. After removal of D, 3-var, and 1-var became the most informative ones. The exponent α lost much of its importance. And again, T_N is the least informative attribute.

TABLE IV. Detailed performance analysis of both classification methods on the test data. Support is the number of trajectories known to belong to a given class. All results are rounded to two decimal digits.

Method	Features	Measure	Normal diffusion	Subdiffusion	Superdiffusion	Total/Average
RF	With D	Support	12 000	12 000	12 000	36 000
		Precision	0.912	0.969	0.966	0.949
		Recall	0.935	0.958	0.951	0.948
		F1	0.923	0.963	0.959	0.948
	No D	Support	12 000	12 000	12 000	36 000
		Precision	0.908	0.967	0.967	0.947
		Recall	0.935	0.955	0.950	0.947
		F1	0.921	0.961	0.958	0.947
GB	With D	Support	12 000	12 000	12 000	36 000
		Precision	0.911	0.967	0.965	0.948
		Recall	0.933	0.958	0.951	0.947
		F1	0.922	0.962	0.958	0.947
	No D	Support	12 000	12 000	12 000	36 000
		Precision	0.907	0.963	0.964	0.945
		Recall	0.928	0.954	0.951	0.944
		F1	0.917	0.958	0.957	0.944

D. Note on auxiliary classifiers

Apart from the main collection of synthetic trajectories described in Sec. V B, we generated two additional training sets. The first one was built from the main set by simply adding noise (Sec. V C) and the second one—with a smaller value (0.38 versus 1) of the parameter σ (Sec. V D).

Those sets were then used to train new classifiers. Their accuracies are listed in Table VI. Note that those values are very similar to the ones presented in Table III. Thus, all classifiers perform very well on their corresponding synthetic test sets. Interestingly, the machine learning algorithms seem to deal excellent with noisy data, as there is no significant drop in the accuracy of the classifiers trained on that data.

The basic characteristics of the additional classifiers turned out to be practically indistinguishable from the ones presented in the previous sections. Thus, we will skip their detailed description for the sake of readability.

TABLE V. Feature importances for both methods, sorted in the descending order of the scores in case of random forest with D. The bold face indicates the most important features in each case. The least important ones are underlined. The "With D" label refers to the full feature set, "No D" one—to the feature set after removal of the diffusion coefficient D.

	Random forest		Gradient boosting	
Feature	With D	No D	With D	No D
2-var	**0.296**	**0.239**	0.238	0.160
α	0.201	0.197	**0.274**	0.125
3-var	0.178	0.183	0.108	**0.245**
1-var	0.171	0.200	0.204	0.210
4-var	0.078	0.110	0.095	0.145
T_N	0.038	$\underline{0.032}$	0.030	$\underline{0.037}$
5-var	0.022	0.038	0.034	0.077
D	$\underline{0.017}$	–	$\underline{0.016}$	–

E. Application to real data

1. Summary of statistical methods

In Table VII, classification results from Ref. [34] for the G protein-coupled receptors and G proteins (see Sec. V A for details) are summarized. Except the standard MSD method, the authors used statistical testing procedures based on: (a) MSD (referred as "MSD test" in Table VII), (b) maximum distance traveled by a particle ("MAX") and (c) p-variations at different values of p ("1-var" and "2-var"). As we can see, the methods do not yield coherent results. MSD classifies most of the trajectories as subdiffusion. The MAX and MSD test procedures indicate a prevalence of freely diffusing particles in the same data set. The p-var tests give similar proportions of normal and subdiffusive trajectories. Moreover, only the standard MSD method is able to recognize a noticeable subset of trajectories as superdiffusion. Further analysis with synthetic data revealed that the p-var method is the most accurate one for FBM, while the MSD/MAX tests are the best choice (in terms of errors) for OU and DBM processes.

2. Classification with full feature set

Our first attempt to classify the data with the whole feature set defined in Sec. VI is presented in Table VIII. As we see, both methods work similarly, with gradient boosting recognizing more G protein trajectories as a superdiffusive motion. Most of the trajectories are classified as normal or subdiffusion, with the prevalence of the latter. Note that the

TABLE VI. Accuracies of the classifiers trained on auxiliary data sets: the first with noise [see Sec. V C and the second with $\sigma = 0.38$ (Sec. V D)].

	Random forest		Gradient boosting	
Data set	With D	No D	With D	No D
With noise	0.946	0.932	0.946	0.930
With $\sigma = 0.38$	0.953	0.950	0.952	0.950

TABLE VII. Summary of the classification results from Ref. [34]. Columns labeled with R and G correspond to the G protein-coupled receptors and G proteins, respectively. The MSD data was calculated for $c = 0.1$ (see Sec. V B for explanation) and the maximum lag equal to 10% of the trajectories' lengths. Due to rounding, the numbers may not add up precisely to 100%.

	MSD		MSD test		MAX		1-var		2-var	
	R	G	R	G	R	G	R	G	R	G
Normal diffusion	19%	22%	79%	76%	79%	76%	53%	52%	47%	51%
Subdiffusion	80%	72%	21%	24%	21%	24%	47%	46%	53%	48%
Superdiffusion	1%	6%	0%	1%	0%	1%	0%	1%	0%	2%

numbers given in Table VIII do not match any of the results generated with the statistical methods. Thus, it is really hard to judge which method should be chosen to work with real data.

3. *Role of D and classification with reduced feature set*

To pin down a possible cause for the deviation from the statistical methods, let us have a look at the distribution of values of the generalized diffusion coefficient D among the trajectories in the real data set. The corresponding histograms for the G protein-coupled receptors and G proteins are shown in Fig. 5. To calculate the histograms, D was simply extracted from the MSD curves under the assumption of the anomalous diffusion model [34]. Its values in the data set turned out to be much smaller than in the synthetic data set, with $D = 0.0715$ being the most frequent one. However, the synthetic training set was generated with $\sigma = 1$ (i.e., $D = 0.5$) for all types of diffusion. Thus, the discrepancy in the classification results from any of the methods presented in [34] may simple be caused by the fact, that the classifiers were trained for a different regime of diffusion.

To check this hypothesis let us classify the trajectories with the models trained on the data set generated with $\sigma = 0.38$, which corresponds to $D = 0.0715$. Classification results with the full set of features are presented in Table IX. The results differ from the previous classification. Now, the shares of the trajectories are very similar to the ones obtained with 1-var and 2-var methods from Ref. [34]. Most of the trajectories belong either to the normal diffusion or the subdiffusion class, with the first having a slightly larger count. There is only a bunch of data samples recognized as superdiffusion in case of G proteins.

Treating the statistical results as a reference point, we can indeed say that adjusting σ to the real data set improved the results. However, this procedure is not really convenient, as it requires generation of a new synthetic data set, extracting of features and training of a new classifier practically every time new experimental samples are arriving.

In search of a more universal procedure we decided to train new classifiers on the reduced set of features not containing the diffusion coefficient D, but we used the main synthetic set with $\sigma = 1$ as in Ref. [34]. Since D turned out to be the least informative feature (Table V), it was the natural candidate for the removal anyway. We know already, that the accuracy of the classifiers without D is a little bit smaller (Table III). Nevertheless, we expect them to work better on unseen data. Indeed, even though the choice of σ was not optimal, the classification results shown in Table X resemble the ones obtained above (Table IX) and the 1-var and 2-var methods from Ref. [34]. Again, most of the trajectories belong either to the normal diffusion or the subdiffusion class, with the first having a larger count. Just few data samples are recognized as superdiffusion in case of G proteins.

The advantage of the classification with the reduced data set over the one with the adjustment of σ lies in that the classifier is trained only once and then may be simply applied to any unseen samples. It does not require a recurring and time consuming procedure of adjustment of σ, generation of tailor-made training data, extraction of features and training of the classifier every time a new set of experimental trajectories is arriving for analysis.

The agreement with the p-variation procedure for small values of p makes perfect sense, if we recall that 2-var and 1-var belong to be the most informative among the features used by the classifiers to distinguish between the data samples.

TABLE VIII. Diffusion modes of real trajectories found with classifiers trained on the main synthetic dataset ($\sigma = 1$, no noise; see Sec. V B) with the full set of features (referred to as "With D" in the previous sections). Due to rounding, the numbers may not add up precisely to 100%.

	Random forest		Gradient boosting	
	Receptor	G protein	Receptor	G protein
Normal diffusion	38%	44%	38%	38%
Subdiffusion	61%	54%	60%	55%
Superdiffusion	0%	1%	0%	5%

TABLE IX. Diffusion types found in real data with the classifiers trained on the auxiliary dataset ($\sigma = 0.38$, no noise; see Sec. V D) with the full set of features (referred to as "With D" in the previous sections). Due to rounding, the numbers may not add up precisely to 100%.

	Random forest		Gradient boosting	
	Receptor	G protein	Receptor	G protein
Normal diffusion	54%	51%	54%	51%
Subdiffusion	45%	47%	45%	46%
Superdiffusion	0%	1%	0%	1%

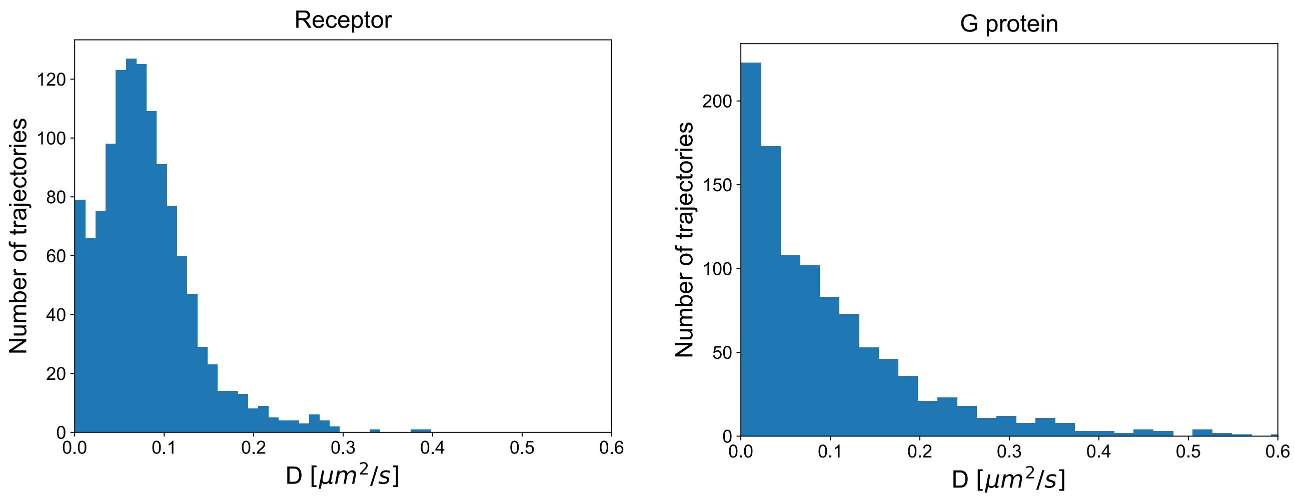

FIG. 5. Distribution of D among trajectories in the real data set.

4. Impact of noise

Although it goes beyond a comparison of machine learning algorithms with the statistical methods from Ref. [34], as its authors used only pure trajectories, we would like to conclude this section by applying the classifier trained on noisy data (Sec. V C) to the real trajectories. Results of classification with the reduced feature set are shown in Table XI.

The introduction of noise has changed the results. Although still most of the trajectories belong either to the normal diffusion or to the subdiffusion class, the first has now larger count compared with the case without noise in the synthetic data. As we already pointed out in Sec. V B, the boundary between the normal and anomalous modes in the vicinity of $\alpha = 1$ is not particularly well defined even in the absence of noise. This boundary is further blurred in the presence of noise, resulting in the observed rearrangement of class memberships. However, if we still would like to relate the "noisy" results with the ones from Ref. [34], then they are close to an average of 1-var and MAX methods.

VIII. DISCUSSION

Machine learning methods used for classification of SPT data are known to sometimes fail to generalize to unseen data [39]. In this paper, we revisited our ML approach to trajectory classification and presented a new set of features, which are required by the classifiers to process the input data. This new set allows the random forest and gradient boosting classifiers to transfer the knowledge from a synthetic training set to real data. The classifiers were tested on a subset of experimental data describing G proteins and G protein-coupled receptors [47]. The results were then compared to four statistical testing procedures introduced in Ref. [34].

We have shown that the choice of the feature set is crucial, as even a small change in its content may significantly impact the behavior of the classifiers. We decided to use a set consisting of the anomalous α exponent, the diffusion coefficient D, the maximum distance traveled by a particle, the power γ^p fitted to p-variation for values of p from 1 to 5. These features were extracted from several statistical methods presented in Ref. [34]. Since none of the methods turned out to be superior to the others, the authors of the work proposed to take a mean of the results of all methods to minimize the risk of large errors. Due to the fact that our classifiers use all features simultaneously as input, in some sense we followed their advice. From our findings it follows that with the full feature set, the machine learning methods applied to the real data yield results completely different from the ones produced with the statistical methods. However, adjusting the diffusion coefficient in the synthetic trajectories to the most frequent value among the real samples or removing this coefficient from the feature set and retraining the classifiers starts to produce results very similar to the p-variation method from Ref. [34].

From the above methods, the one with the reduced set of features is more convenient, because the classifier is trained

TABLE X. Results of classifiers trained on the main synthetic dataset ($\sigma = 1$, no noise) with the reduced set of features, after removal of D (referred to as "No D" in the previous sections). Due to rounding, the numbers may not add up precisely to 100%.

	Random forest		Gradient boosting	
	Receptor	G protein	Receptor	G protein
Normal diffusion	52%	53%	56%	54%
Subdiffusion	46%	45%	43%	43%
Superdiffusion	0%	1%	0%	1%

TABLE XI. Results of classifiers trained on noisy data ($\sigma = 1$) with the reduced set of features (referred to as "No D" in the previous sections). Due to rounding, the numbers may not add up precisely to 100%.

	Random forest		Gradient boosting	
	Receptor	G protein	Receptor	G protein
Normal diffusion	62%	58%	63%	58%
Subdiffusion	38%	40%	36%	40%
Superdiffusion	0%	2%	0%	3%

only once and then may be simply applied to any unseen data. It does not require a recurring and time consuming procedure consisting of: (1) adjustment of σ, (2) generation of tailor-made training data, (3) extraction of features, and (4) training of the classifier every time a new set of experimental trajectories is arriving for analysis.

The agreement between the ML approach and the statistical testing based on p variations is, on the one hand, a confirmation that our ML methods are able to classify unseen data in a reasonable way. On the other hand, it may support the choice of the p-variation testing procedure among the statistical methods.

Introduction of noise mimicking different kinds of localization errors changed the classification results—the count of normal diffusion (subdiffusion) trajectories increased (decreased) by a couple of percentage points. A slight increase of superdiffusion samples was also observed in case of G proteins. If we would like to relate the "noisy" results with the ones from Ref. [34], then they are close to an average of 1-var and MAX methods.

Although still a lot needs to be done in terms of selection of robust features and the generation of appropriate synthetic training data, we believe that our methodology may be successfully applied to experimental data to provide a further insight into the dynamics of complex biological processes.

ACKNOWLEDGMENTS

The work of P.K., H.L.-O., J.S., and A.W. was supported by NCN-DFG Beethoven Grant No. 2016/23/G/ST1/04083. The work of J.J. was supported by NCN Sonata Bis Grant No. 2019/34/E/ST1/00360. Calculations have been carried out using resources provided by Wroclaw Centre for Networking and Supercomputing (http://wcss.pl).

APPENDIX: CODES

Python codes for every stage of the classification procedure shown in Fig. 1, together with a short documentation, are publicly available at Zenodo (see Ref. [93]).

[1] L. Barak and W. Webb, J. Cell Biol. **95**, 846 (1982).
[2] A. Kusumi, Y. Sako, and M. Yamamoto, Biophys. J. **65**, 2021 (1993).
[3] A. Yildiz, J. N. Forkey, S. A. McKinney, T. Ha, Y. E. Goldman, and P. R. Selvin, Science **300**, 2061 (2003).
[4] C. Kural, H. Kim, S. Syed, G. Goshima, V. I. Gelfand, and P. R. Selvin, Science **308**, 1469 (2005).
[5] I. Izeddin, V. Récamier, L. Bosanac, I. I. Cissé, L. Boudarene, C. Dugast-Darzacq, F. Proux, O. Bénichou, R. Voituriez, O. Bensaude, M. Dahan, and X. Darzacq, eLife **3**, e02230 (2014).
[6] N. Gal, D. Lechtman-Goldstein, and D. Weihs, Rheol. Acta **52**, 425 (2013).
[7] P. C. Bressloff, *Stochastic Processes in Cell Biology*, Interdisciplinary applied mathematics (Springer, Cham, 2014), pp. 645–672.
[8] M. J. Saxton, Biophys. J. **67**, 2110 (1994).
[9] M. J. Saxton and K. Jacobson, Annu. Rev. Biophys. Biomol. Struct. **26**, 373 (1997).
[10] T. J. Feder, I. Brust-Mascher, J. P. Slattery, B. Baird, and W. W. Webb, Biophys. J. **70**, 2767 (1996).
[11] R. Metzler and J. Klafter, Phys. Rep. **339**, 1 (2000).
[12] X. Michalet, Phys. Rev. E **82**, 041914 (2010).
[13] S. B. Alves, G. F. O. Jr., L. C. Oliveira, T. P. de Silansa, M. Chevrollier, M. Oriá, and H. L. S. Cavalcante, Physica A **447**, 392 (2016).
[14] N. Hoze, D. Nair, E. Hosy, C. Sieben, S. Manley, A. Herrmann, J.-B. Sibarita, D. Choquet, and D. Holcman, Proc. Natl. Acad. Sci. USA **109**, 17052 (2012).
[15] H. Berry and H. Chaté, Phys. Rev. E **89**, 022708 (2014).
[16] M. Weiss, M. Elsner, F. Kartberg, and T. Nilsson, Biophys. J. **87**, 3518 (2004).
[17] D. Arcizet, B. Meier, E. Sackmann, J. O. Rädler, and D. Heinrich, Phys. Rev. Lett. **101**, 248103 (2008).
[18] N. Monnier, S.-M. Guo, M. Mori, J. He, P. Lénárt, and M. Bathe, Biophys. J. **103**, 616 (2012).
[19] E. Kepten, A. Weron, G. Sikora, K. Burnecki, and Y. Garini, PLoS One **10** (2), e0117722 (2015).
[20] V. Briane, C. Kervrann, and M. Vimond, Phys. Rev. E **97**, 062121 (2018).
[21] M. Weiss, Phys. Rev. E **100**, 042125 (2019).
[22] M. J. Saxton, Biophys. J. **64**, 1766 (1993).
[23] M. T. Valentine, P. D. Kaplan, D. Thota, J. C. Crocker, T. Gisler, R. K. Prud'homme, M. Beck, and D. A. Weitz, Phys. Rev. E **64**, 061506 (2001).
[24] N. Gal and D. Weihs, Phys. Rev. E **81**, 020903(R) (2010).
[25] D. S. Grebenkov, M. Vahabi, E. Bertseva, L. Forró, and S. Jeney, Phys. Rev. E **88**, 040701(R) (2013).
[26] A. Fuliński, J. Phys. A: Math. Theor. **50**, 054002 (2017).
[27] C. Raupach, D. P. Zitterbart, C. T. Mierke, C. Metzner, F. A. Müller, and B. Fabry, Phys. Rev. E **76**, 011918 (2007).
[28] S. Burov, S. M. A. Tabei, T. Huynh, M. P. Murrell, L. H. Philipson, S. A. Rice, M. L. Gardel, N. F. Scherer, and A. R. Dinner, Proc. Natl. Acad. Sci. USA **110**, 19689 (2013).
[29] V. Tejedor, O. Bénichou, R. Voituriez, R. Jungmann, F. Simmel, C. Selhuber-Unkel, L. B. Oddershede, and R. Metzler, Biophys. J. **98**, 1364 (2010).
[30] K. Burnecki, E. Kepten, Y. Garini, G. Sikora, and A. Weron, Sci. Rep. **5**, 11306 (2015).
[31] R. Das, C. W. Cairo, and D. Coombs, PLoS Comput. Biol. **5**, 1 (2009).
[32] P. J. Slator, C. W. Cairo, and N. J. Burroughs, PLOS One **10**, e0140759 (2015).
[33] P. J. Slator and N. J. Burroughs, Biophys. J. **115**, 1741 (2018).
[34] A. Weron, J. Janczura, E. Boryczka, T. Sungkaworn, and D. Calebiro, Phys. Rev. E **99**, 042149 (2019).
[35] S. Raschka, *Python Machine Learning* (Packt Publishing, Birmingham, UK, 2015).
[36] S. Thapa, M. A. Lomholt, J. Krog, A. G. Cherstvy, and R. Metzler, Phys. Chem. Chem. Phys. **20**, 29018 (2018).
[37] A. G. Cherstvy, S. Thapa, C. E. Wagner, and R. Metzler, Soft Matter **15**, 2526 (2019).

[38] T. Wagner, A. Kroll, C. R. Haramagatti, H.-G. Lipinski, and M. Wiemann, PLoS One **12** (1), e0170165 (2017).
[39] P. Kowalek, H. Loch-Olszewska, and J. Szwabiński, Phys. Rev. E **100**, 032410 (2019).
[40] G. Muñoz-Gil, M. A. Garcia-March, C. Manzo, J. D. Martín-Guerrero, and M. Lewenstein, New J. Phys. **22**, 013010 (2020).
[41] P. Dosset, P. Rassam, L. Fernandez, C. Espenel, E. Rubinstein, E. Margeat, and P.-E. Milhiet, BMC Bioinform. **17**, 197 (2016).
[42] N. Granik, L. E. Weiss, E. Nehme, M. Levin, M. Chein, E. Perlson, Y. Roichman, and Y. Shechtman, Biophys. J. **117**, 185 (2019).
[43] S. Bo, F. Schmidt, R. Eichhorn, and G. Volpe, Phys. Rev. E **100**, 010102(R) (2019).
[44] D. Han, N. Korabel, R. Chen, M. Johnston, A. Gavrilova, V. J. Allan, S. Fedotov, and T. A. Waigh, eLife **9**, e52224 (2020).
[45] N. Hatami, Y. Gavet, and J. Debayle, in *Proceedings of the SPIE 10th International Conference on Machine Vision (ICMV'17)*, edited by A. Verikas, P. Radeva, D. Nikolaev, and J. Zhou (SPIE, Bellingham, Washington, USA, 2018), p. 10696.
[46] T. Mitchel, *Machine Learning* (McGraw-Hill Professional, New York, 1997).
[47] T. Sungkaworn, M.-L. Jobin, K. Burnecki, A. Weron, M. J. Lohse, and D. Calebiro, Nature **550**, 543 (2017).
[48] F. Höfling and T. Franosch, Rep. Prog. Phys. **76**, 046602 (2013).
[49] J. Szymanski and M. Weiss, Phys. Rev. Lett. **103**, 038102 (2009).
[50] J.-H. Jeon and R. Metzler, Phys. Rev. E **81**, 021103 (2010).
[51] A. V. Weigel, B. Simon, M. M. Tamkun, and D. Krapf, Proc. Natl. Acad. Sci. USA **108**, 6438 (2011).
[52] A. V. Weigel, S. Ragi, M. L. Reid, E. K. P. Chong, M. M. Tamkun, and D. Krapf, Phys. Rev. E **85**, 041924 (2012).
[53] M. Hellmann, J. Klafter, D. W. Heermann, and M. Weiss, J. Phys.: Condens. Matter **23**, 234113 (2011).
[54] S. Sadegh, J. L. Higgins, P. C. Mannion, M. M. Tamkun, and D. Krapf, Phys. Rev. X **7**, 011031 (2017).
[55] H. Qian, M. P. Sheetz, and E. L. Elson, Biophys. J. **60**, 910 (1991).
[56] A. R. Vega, S. A. Freeman, S. Grinstein, and K. Jaqaman, Biophys. J. **114**, 1018 (2018).
[57] Y. Lanoiselée, G. Sikora, A. Grzesiek, D. S. Grebenkov, and A. Wyłomańska, Phys. Rev. E **98**, 062139 (2018).
[58] T. K. Ho, in *Proceedings of the 3rd International Conference on Document Analysis and Recognition—Volume 1* (IEEE Computer Society, Washungton, D.C., 1995).
[59] T. K. Ho, IEEE Trans. Pattern Anal. Mach. Intell. **20** (8), 832 (1998).
[60] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, Ann. Stat. **26**, 1651 (1998).
[61] J. H. Friedman, Comput. Stat. Data Anal. **38**, 367 (2002).
[62] Y.-Y. Song and Y. LU, Shanghai Arch. Psych. **27**, 130 (2015).
[63] C. E. Shannon, Bell Syst. Tech. J. **27**, 379 (1948).
[64] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning with Applications in R* (Springer, New York, NY, 2013).
[65] M. Bramer, *Principles of Data Mining*, 2nd ed. (Springer, Berlin, 2013).
[66] S. Havlin and D. Ben-Avraham, Adv. Phys. **36**, 695 (1987).
[67] O. Blondel, M. R. Hilário, R. S. dos Santos, V. Sidoravicius, and A. Teixeira, Electron. J. Probab. **24**, 33 (2019).
[68] J. P. Straley, J. Phys. C: Solid State Phys. **13**, 2991 (1980).
[69] R. Metzler, J.-H. Jeon, A. G. Cherstvy, and E. Barkai, Phys. Chem. Chem. Phys. **16**, 24128 (2014).
[70] B. B. Mandelbrot and J. W. V. Ness, SIAM Rev. **10**, 422 (1968).
[71] G. Guigas, C. Kalla, and M. Weiss, Biophys. J. **93**, 316 (2007).
[72] K. Burnecki, E. Kepten, J. Janczura, I. Bronshtein, Y. Garini, and A. Weron, Biophys. J. **103**, 1839 (2012).
[73] K. Burnecki and A. Weron, Phys. Rev. E **82**, 021130 (2010).
[74] S. C. Kou and X. S. Xie, Phys. Rev. Lett. **93**, 180603 (2004).
[75] K. Burnecki and A. Weron, J. Stat. Mech. (2014) P10036.
[76] T. C. Elston, J. Math. Biol. **41**, 189 (2000).
[77] C. L. MacLeod, Z. Ivezi, C. S. Kochanek, S. Kozłowski, B. Kelly, E. Bullock, A. Kimball, B. Sesar, D. Westman, K. Brooks, R. Gibson, A. C. Becker, and W. H. de Vries, Astrophys. J. **721**, 1014 (2010).
[78] J.-H. Jeon, V. Tejedor, S. Burov, E. Barkai, C. Selhuber-Unkel, K. Berg-Sørensen, L. Oddershede, and R. Metzler, Phys. Rev. Lett. **106**, 048103 (2011).
[79] K. L. Pierce, R. T. Premont, and R. J. Lefkowitz, Nat. Rev. Mol. Cell Biol. **3**, 639 (2002).
[80] C. Flynn, FBM: Exact methods for simulating fractional Brownian motion (FBM) or fractional Gaussian noise (FGN) in Python (2017), retrieved from https://github.com/crflynn/fbm.
[81] R. B. Davies and D. S. Harte, Biometrika **74**, 95 (1987).
[82] J. Hosking, Water Resour. Res. **20**, 1898 (1984).
[83] C. Flynn, Stochastic: A python package for generating realizations of stochastic processes (2018), retrieved from https://github.com/crflynn/stochastic/.
[84] P. E. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations* (Springer-Verlag, Berlin, 1992).
[85] J. W. R. Mellnik, M. Lysy, P. A. Vasquez, N. S. Pillai, D. B. Hill, J. Cribb, S. A. McKinley, and M. G. Forest, J. Rheol. **60**, 379 (2016).
[86] H. Deschout, F. C. Zanacchi, M. Mlodzianoski, A. Diaspro, J. Bewersdorf, S. T. Hess, and K. Braeckmans, Nat. Methods **11**, 253 (2014).
[87] M. Weiss, Phys. Rev. E **88**, 010101(R) (2013).
[88] C. P. Calderon, Phys. Rev. E **93**, 053303 (2016).
[89] M. Magdziarz, A. Weron, K. Burnecki, and J. Klafter, Phys. Rev. Lett. **103**, 180602 (2009).
[90] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, J. Mach. Learn. Res. **12**, 2825 (2011).
[91] J. W. Perry, A. Kent, and M. M. Berry, Am. Doc. **6**, 242 (1955).
[92] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees* (Wadsworth and Brooks, Monterey, CA, 1984).
[93] J. Janczura, P. Kowalek, H. Loch-Olszewska, J. Szwabiński, and A. Weron, Trajectory generators and classificators for fractional anomalous diffusion classification (2020), retrieved from https://doi.org/10.5281/zenodo.3933357.