**Machine learning application of Gaussian Processes and Reproducing Kernel Hilbert Space (RKHS) to the task of comparing functional state molecular dynamics (MD) of biological macromolecules**

NOTE: provides alternative to KL divergence in DROIDS and the multi-agent learner in maxDemon

<https://gbabbitt.github.io/DROIDS-4.0-comparative-protein-dynamics/>

Intro to RKHS and its application to defining a distance (i.e. difference in means) in feature vector space)

<https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.705.292&rep=rep1&type=pdf>

Note: see section 5 Application to distances between means in feature space (MMD - maximum mean discrepancy)

Overview- Our past projects (DROIDS/maxDemon Linux pipeline) will be moved to a webserver (client side = js, web assembly or wasm) and server side (python science stack). The webserver will allow users to conduct molecular dynamics simulations with any engine they prefer (NAMD, Amber, Gormacs/CHARMM, OpenMM) prior to running our software. The webserver will employ client side compiling of cpptraj (AmberTools) via javascript web assembly in order to repeatedly sample and extract trajectory analyses from binary files (.nc) that would otherwise be too large to move over a network (i.e. these are typically terabyte scale). As molecular bond vibrations tend to exhibit largely Gaussian behavior, we will also test and employ a new unifying machine learning approach to all stages of the general problem of comparing simulations collected from different functional protein states. This will move us away from the KL divergence metric towards a Bayesian probabilistic learning method described below. We will also move away from the multi-agent learner for identifying functionally conserved dynamics regions of a protein and move towards a Bayesian classifier based on Gaussian modeling that can be used to describe mean distances and conduct hypothesis testing on the learned feature spaces in our two-state functional protein dynamics comparisons. The end goal here is to unify all comparative tasks under one machine learning framework (Gaussian processes and kernel machines) and to improve speed, reliability, interpretability and ease of use at the same time. The various steps in our plan are outlined below.

1. Defining the initial feature space on a molecular dynamics (MD) simulation

After MD simulation (conducted locally on the user’s computer), our webserver will compile and repeatedly subsample a MD trajectory by running cpptraj functions from AmberTools on the client side to calculate atomic fluctuation (flux) and atomic correlation (corr) and construct two sub-feature vectors for the machine learning. These features are dynamics measures that are collected both (A) within a given amino acid site on the protein and (B) across all sites on the protein. The specific atoms involved in the calculations are the backbone amino acid atoms N, C, Cα, and O. Time slices of the MD simulation are denoted as t1, t2, t3 etc.

1. Sub-feature vector for molecular dynamics within a given site

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Nflux | Cαflux | Cflux | Oflux | Cα-Ncorr | Cα-Ccorr | Cα-Ocorr |
| t1 |  |  |  |  |  |  |  |
| t2 |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |
| tn |  |  |  |  |  |  |  |

1. Sub-feature vector for molecular dynamics interacting across sites (centered on alpha carbons)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Cαflux | Cα-Cαcorr(site1) | Cα-Cαcorr(site2) | … | Cα-Cαcorr(siteN) |
| t1 |  |  |  |  |  |
| t2 |  |  |  |  |  |
| … |  |  |  |  |  |
| tn |  |  |  |  |  |

1. Defining a kernel for the Gaussian process classifier (GPC) and related reproducing kernel Hilbert space (RKHS)

The kernel functions available for a Gaussian Process Classifier (GPC) in the python sci-kit learn library will be compared on random subsets of the existing training data via a 5-fold cross validation and the one with the highest performance will be retained for classification learning and also used to define the reproducing kernel Hilbert space (RKHS) associated with that kernel function. NOTE: if n = number of time slices and p = number of features, then the cross-validation subsets must satisfy that n/p>5. A site-wise GPC with the best tuned candidate kernel function will then be employed to define a Bayesian probabilistic boundary between the two protein dynamic states at each amino acid site. Candidate kernels will include ConstantKernel, RBF, Matern, RationalQuadratric, ExpSineSquared and DotProduct. All candidate kernel functions will be tested individually AND as sum kernels with white noise (e.g. RBF + WhiteKernel). Some other rational combinations of sum, product and exponentiation kernels will also be tested as well. NOTE: all candidate kernels must satisfy the constraints of RKHS.

<https://scikit-learn.org/stable/modules/gaussian_process.html>

<https://scikit-learn.org/stable/auto_examples/gaussian_process/plot_gpc_xor.html#sphx-glr-auto-examples-gaussian-process-plot-gpc-xor-py>

1. Defining the final full feature vector for site-wise molecular dynamics

The correlations between most pairs of sites across the protein will be negligible as one moves away along the protein backbone from the reference site unless the tertiary folding brings distant pairs of sites in proximity in 3-D space. Therefore, the information regarding site interactions in this sub-feature matrix can be easily reduced using a kernel principal components analysis (kernel PCA). The strongest PC’s representing most of the covariance between potential sites will be kept and PC scores for these main variance components for each time slice will be retained and added to the sub-feature vector for the dynamics within the sites.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Nflux | Cαflux | Cflux | Oflux | Cα-Ncorr | Cα-Ccorr | Cα-Ocorr | PC1 | PC2 | … | PC(N) |
| t1 |  |  |  |  |  |  |  |  |  |  |  |
| t2 |  |  |  |  |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |  |  |  |  |
| tn |  |  |  |  |  |  |  |  |  |  |  |

1. Utilizing RKHS to define significant site-wise distances in molecular dynamics defining two functional protein states

The RKHS defined above will be used to define the distances between the molecular dynamics of two functional states of a protein (i.e. two molecular dynamics simulations representing differences caused by binding interactions involving chemical signals, toxins or drugs, or differences caused by genetic and epigenetic changes). This will be defined by the maximum mean discrepancy (MMD) of the pre-classifications represented in RKHS. This will replace the use of divergence metrics (mean difference and Kullback-Leibler or KL divergence) in our previous work. The MMD’s of single sites can be locally color-mapped to protein structure, indicating important regions of functional changes in atom motions at specific sites, while a protein-wide MMD can be compared to measurable properties of proteins from the lab (i.e. binding energies, kinetic properties, thermostability etc. ). Two sample hypothesis tests can be also conducted upon on MMD (maximum mean discrepancy) defined by RKHS (see second video below). This will replace our use of two-sample Kolmogorov-Smirnov (KS) tests in our past work.

Arthur Gretton’s lectures on RKHS and applications

<https://www.youtube.com/watch?v=alrKls6BORc>

<https://www.youtube.com/watch?v=eANiXrWO1dM>

easy-to-understand blog with python code for RKHS

<https://hpccsystems.com/blog/reproducing-RKHS>

1. Identifying functional from non-functional dynamics (signal vs. noise) using a Gaussian Process Classifier (GPC).

The site-wise Gaussian Process Classifier (GPC) trained on the two-state dynamics generated by the user for comparison can also be deployed on N unclassified simulations representing K orthologs. Then canonical correlations in performance metrics can be used to map regions with significantly conserved dynamics (i.e. the ability of the learner to correctly identify local functional dynamics classes within N simulations locally correlated over K evolutionary ortholog proteins). The K ortholog sets of N new simulations in each of the two functional dynamic states (e.g. bound vs. unbound) will be used as unlabeled test data sets for a Bayesian probabilistic classifier that will predict the dynamic state class (e.g. bound=1 or unbound=0) for the unlabeled simulations. Learning performance profiles for each site/position will be defined as the average correct classification over 2N simulations (y axis) over all given sites (x axis). Note: inability to learn will produce a profile close to = 0.5. A local correlation or association in learning performance profiles over K ortholog sets can be used to identify local regions of evolutionary (i.e. functionally) conserved dynamics and subsequently mapped on the structure of the protein. A key assumption here is that dynamics due to random thermal noise will never repeat in the same way in N simulations on K orthologs, therefore the correlation in learning performance can be used to identify non-random classification patterns indicative of functional dynamics of the protein. Further information theoretics applied to these classifications can be employed to compare genetic/drug class variants (e.g. relative entropy) AND coordinated dynamics (e.g. mutual information). See our past publications for more details. NOTE: this follows the same logic as our past iteration of maxDemon 2.0 but replaces the multi-agent learner with multiple orthologous iterations of our best candidate GPC. Ideally, k ortholog structures will be obtained from the PDB, but if not they can be modelled from sequence alignment using our ENTRÉE project code.

<https://github.com/gbabbitt/EnTree>

additional resources on the machine learning approach

<https://scikit-learn.org/stable/modules/gaussian_process.html#kernels-for-gaussian-processes>

<https://machinelearningmastery.com/gaussian-processes-for-classification-with-python/>

<http://krasserm.github.io/2020/11/04/gaussian-processes-classification/>

<https://towardsdatascience.com/kernel-machine-from-scratch-718eba74ea3e>

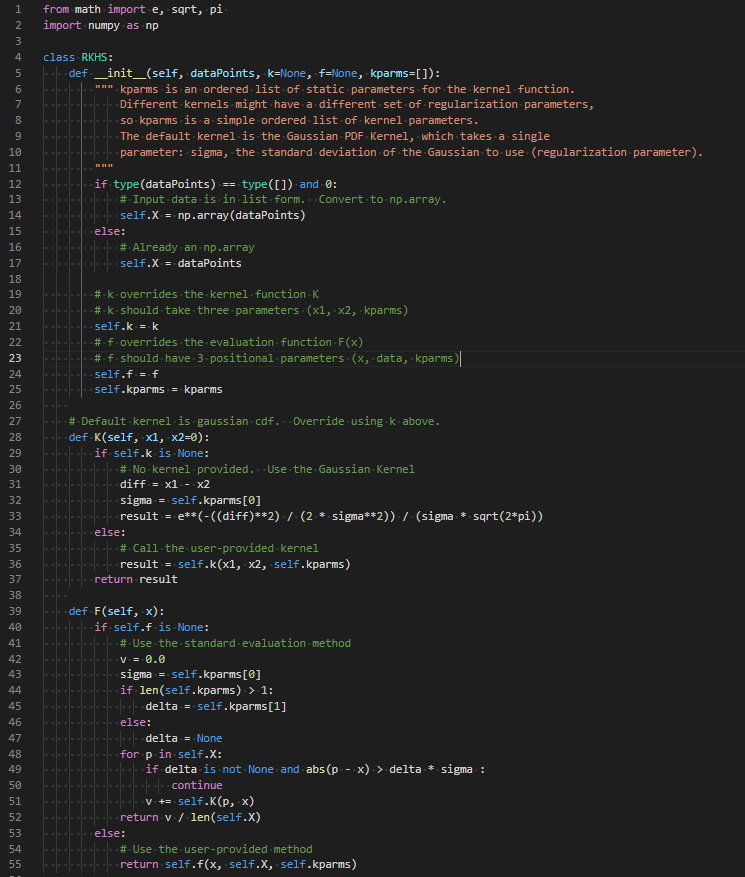
additional resources on the webserver construction and cpptraj

<https://developer.mozilla.org/en-US/docs/WebAssembly/Using_the_JavaScript_API>

<https://wasmbyexample.dev/home.en-us.html>

<https://amberhub.chpc.utah.edu/cpptraj/>

Generalized RKHS in python



sample code (function plotting)

Chart, histogram

Description automatically generated

Text

Description automatically generated