Implementation of a Long Short-Term Memory Neural Network for the Selection of Relevant Collective Variables on Dynamical Systems

University College London

Department of Physics and Astronomy

Scientific and Data Intensive Computing Individual Research Project

A dissertation submitted for the degree of

Master of Science

of

University College London

Juexi Shao

Dissertation submission date: September 8, 2022

I, Shao Juexi, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work. Link of the GitHub repository: https://github.com/handsomeshao11/PHAS0077_code_repo

Abstract

The analysis of ligand unbinding is helpful for the discovery of new medicines for the treatment of diseases. Biomolecular simulation systems can be used on the study of ligand binding/unbinding. Innovations in dynamics and machine learning have advanced the development of ligand unbinding simulations. In this dissertation, we model the data for protein disassociation and generate features from the data by combinations of linear (1D) and non-linear (2D).Then we propose a Long Short-Term Memory Neural Network to predict the simulation process of ligand unbinding, and uses Machine Learning Transition State Analysis(MLTSA) and Local interpretable model-agnostic explanations(LIME) to find the key features in the simulation process. In addition, there are comparisons among Multilayer perceptrons (MLP), LSTM and some alternatives.

Acknowledgements

First of all, I am sincerely grateful to my supervisor Edina Rosta for giving me a great explanation of the project, and offering an opportunity to study and research in the laboratory. With her guidance, I had deep insight into the project, and set a clear goal.

Second, I would like to convey my gratitude to my co-supervisor Pedro Buigues, who never tires of helping me to understand the knowledge of biology and chemistry and to solve technical confusion I met. He was significant to the advancement of the entire project.

Finally, I am grateful to my department for providing me with the opportunity to develop my interest in learning.

Contents

Chapter 1 **Introduction**

This chapter introduces the background of significance of ligand unbinding and transition state in the domain of biochemistry, and mentions some methods combined dynamical system simulations and machine learning which explores ligand unbinding simulations.

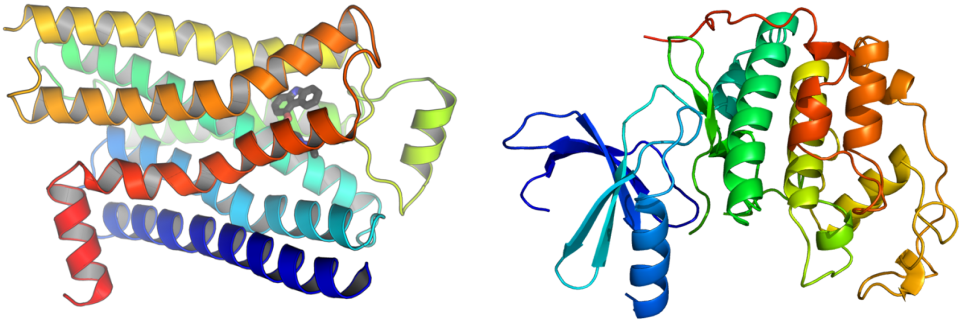
1.1 General Background

1.1.1 Background

Ligand binding in biochemistry and pharmacology may be used to explain the process of receptor and inhibitor assembly. Ligand is a substance that forms complexes with biomolecules for biological purposes. Binding occurs through intermolecular forces, such as ionic bonds, hydrogen bonds, and van der Waals forces.[1] Conversely, unbinding represents the separation of ligand and biomolecules. In the unbiniding process, transition state (TS) could be used to describe a state with the highest energy, along the reaction coordinate.[2] Therefore, in pharmacology, transition state (TS) are widely applied on the development of antibodies for some diseases.[3]

To describe the intermolecular forces in the transition state, as one type of reaction coordinate, collective variables are introduced to be a measure index of the composition of the reaction system in dynamics simulations.[4] There are some receptors used for pharmacology research, such as M3-GPCR and CDK2.

G protein-coupled receptors (GPCRs) (see figure 1.1(a)) could adjust some essential physiological functions, which are known as one of the most necessary therapeutic targets for a series of illness.[5] Receptor pharmacology provides new way for GPCRs drug discovery, combined with some advances in structural biology and innovations in biotechnology.[6]



(a) (b)

Figure 1.1: The structure of GPCR (a) and the CDK2 (b)

An enzyme exists in the human body encoded by the Cyclin-dependent kinase 2 gene, also known as cell division protein kinase 2, or CDK2(see figure 1.1(b)).[7] In cells with normal functions, CDK2 is not essential in the cell cycle. However, CDK2 plays a vital role in the abnormal growth processes of cancer cells.[8] Inhibitors of CDK2 are used to limit the growth of cancer cells. Pre-clinical models have presented initial success in limiting tumor growth, and a reduction in the side effects of current chemotherapy drugs has also been observed.[9][10][11]

1.1.2 Problem Statement

In fact, for experimental determination of ligand kinetics, ligand synthesis is essentially in demand, which could require a high quantity of funding and also be time-expensive, even for moderate numbers of compounds.[12] Accordingly, finding a efficient and reliable way to predict the kinetics of ligand-protein interactions is a important challenge in discovering drug.[13] Apart from this, how to find the key CVs in the process of ligand unbinding is another knotty problem. And some approaches using machine learning method to predict CVs on dynamics simulations are exploring recently.

1.2 Research Gap

The simulation of biomolecular systems is in the stage of rapid development, and many dynamical systems such as MD,[14] SMD,[15], and langevin dynamics,[16] have played an important role on that. In addition, recently, due to a strong predictive ability, some machine learning method like neural networks (NNs) emerged in some researches exploring protein dynam-

ics simulation.[17] And MLP, GBDT have shown good results in the simulation of ligand unbinding.[12]

1.3 Aims and Objectives

1.3.1 Aims

Improve the Machine Learning Transition State Analysis (MLTSA) with a novel ML architecture (LSTM,Stacked LSTM, and bidirectional LSTM) to consider kinetics of the dynamics simulation data, which is helpful to find relevant collective variables to aid in Drug Discovery (like CDK2 inhibitors and M3-GPCR).

1.3.2 Objectives

The deliverable of the whole project code are individed into two sections, which are final transition state predictions with LSTM and other models, important features detecting with Machine Learning Transition State Analysis (MLTSA) and Local interpretable model-agnostic explanations (LIME). In this dissertation, the theory of these two sections are concluded as the methodology, and for the result of these sections, would be explained with emphasis.

1.3.3 Definitions of Key Terms

Collective variables (CVs): An definition in the study of highdimensional dynamical systems, molecular dynamics of macromolecules, as well as liquids, and polymers, especially represents relevant metastable states and state-transition or phasetransition.[18]

Molecular dynamics (MD): A code tool with great computational function to understand the behavior of biological processes, like protein-ligand interactions, at the atomic level.[19]

Transition state (TS): A specific configuration along the coordinates of the reaction in the field of chemistry. It is defined as the state corresponding to the highest potential energy along that reaction coordinate.[2]

Long short-term memory (LSTM): An artificial neural network with feedback connection used in deep learning, which can process both single data points, like images, and also sequential data.[20] In the project, the data generator can simulate the unbinding process and produce a

sequence of data with a specific time span. LSTM could process the data effectively and create correlations among times.

Multilayer perceptron (MLP): A classical neural network consists of input, hidden and output layers, process information forward and calculate error by backpropagation algorithm.[21]

1.4 Structure of Dissertation

The second chapter would elaborate the past literature, including ligand unbindig research developments, dynamics on molecular system developments, neural networks developments, and machine learning on Dynamics on molecular systems.

The third chapter would state works of the data generator (both 1D and 2D) concisely, explain the implementation of the LSTM prediction functions, introduce how to analyze the importance ranking of features in the unbinding simulation by MLTSA and LIME.

The fourth chapter mainly focus on the evaluation and discussion of LSTM model, compared with other models (such as Stacked LSTM, Bidirectional LSTM, MLP), and the important features detecting with MLTSA and LIME.

The fifth chapter contains the summary of evaluating machine learning models (LSTMs and MLP), a feedback of advantages, limitations and some recommendations about the future works.

Chapter 2 **Literature Review**

In this chapter, researches in ligand binding/unbinding, dynamics on molecular systems, neural networks and machine learning on molecular dynamics, in past decades would be elaborated.

2.1 Ligand binding(unbinding) research developments

In the field of clinical medicine, for some molecules like CDK2 and M3-GPCR, finding effective inhibitors are highly concerned recently. In progress of drug design, Robert A. Copeland and his group declared the critical factor of in vivo pharmacological activity and duration is the lifetime or residence time of the binary drug-target complex, rather than the binding affinity between the drug and the target,in 2006.[22] From that, a series of models of clinical significance are designed. Robert A. Copeland created a drug–target residence time model in 2006 and improved it over years.[22] The drug–target residence time model have been proved a crucial significance in numerous clinical drugs, including inhibitors of the chromatin remodelling histone methyltransferases DOT1L, EZH2, and the HIV protease, as well as the HCV protease NS3-4A .[23][24][25] In 2015, particular mathematical model was proposed by Walkupv, to calculate the PAE for a number of inhibitors of the Pseudomonas aeruginosa enzyme target LpxC.[26]

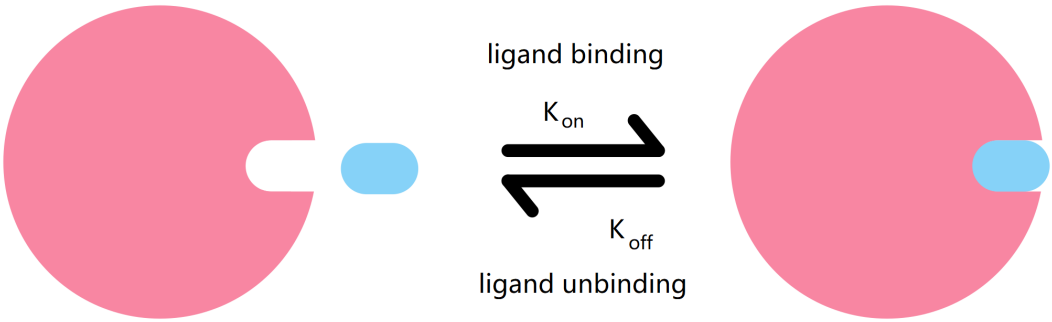


Figure 2.1: Ligand binding and unbinding

In biochemistry and pharmacology, ligand unbinding is the term defined by the process of molecule and ligand dissociation (see figure 2.1).

Transition state (TS) (see figure2.2) exists in the process of the reaction between molecules, for example, ligand and protein. The transition state theory explains that there is a dynamical equilibrium between reactants and activated transition state complexes.[28]

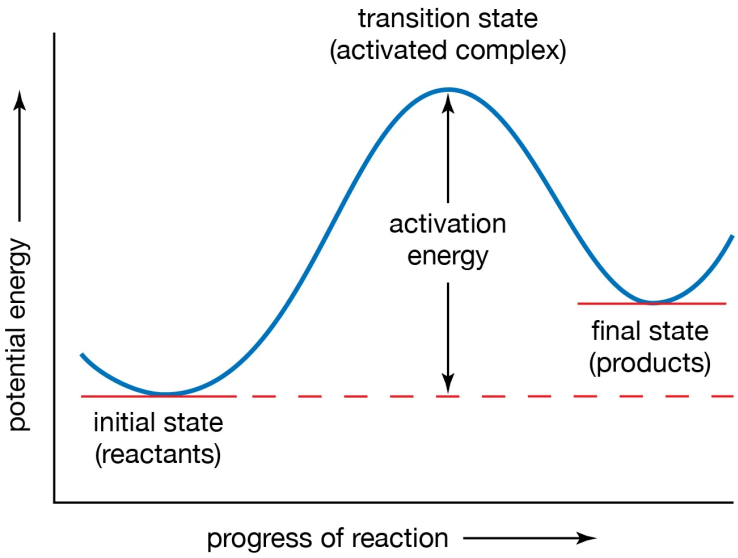


Figure 2.2: Transition state in the reaction progress.[27]

Due to a massive funding and time costing, for experimental determination of ligand kinetics, the demand of ligand synthesis is difficult to be satisfied, even for moderate numbers of compounds.[12] To research the kinetics of ligand-receptor unbinding, a lot of approaches has been provided over decades, divided roughly into Radioligands, Surface Plasmon Resonance Spectroscopy, Molecular Dynamics and etc.[14] Radioligand binding assay, the one of the first radioligands techniques applied for the biochemical, characterisation of the forming of specific drug-receptor complexes.[29] To investigate a radioligand binding in homogeneous system, there is method called Scintillation proximity assay (SPA), which could skip the separate non-bound components separation step.[30] Nevertheless, no further research of SPA was found in radioligand binding kinetic.[14] From the research direction of Surface Plasmon Resonance Spectroscopy, it is a optical technology calculating the refractive index variation near a sensor surface.[31] SPA could quantify data of protein–ligand binding kinetics and affinities real-time, but evaluate an unstable association rate because of protein immobilization.[31][32] To predict absolute kinetic arguments of binding/unbinding, a computational technique called Molecular Dynamics (MD), based on Newtonian mechanics, was attempted to simulate the binding/unbinding kinetics.[14] Besides, there are also other methods used on the molecular systems.

2.2 Dynamics on molecular system developments

Molecular dynamics is calculation tool with great computational function to understand the behavior of biological processes, like protein-ligand interactions.[19] However, a traditional MD system have the disadvantages in controlling time range, which means computer can not generate a standard timescale data as a standard slow biomolecular processes.[14] Therefore, to simulate a more accurate biological kinetic process and simpling better, a series of optimized MD method have been found through years.

Steered Molecular Dynamics (SMD) was proved that it is effective in the calculation of the unbinding free energy profile for TAK-632 and PLX4720 bound to B-RAF.[15] Metadynamics, as well as MetaD (or MTD) was used to predict ligandprotein unbinding of p38 MAP kinase bound to type II inhibitors.[33] Integrated with quantum-mechanical (QM) and molecular mechanical methods (MM), MetaD got more accurate result in kinetic simulation problem.[34] In addition, Collective Variables (CVs), which can describe the intermolecular forces, are widely used in SMD,[35], MetaD,[36] and other dynamics methods.

Besides, Langevin dynamics, developed by Paul Langevin, is a mathematical modelling method applied on the molecular system,[37] and it is one type of Monte Carlo simulation.[38] Combined with simplified models and stochastic differential equations, Langevin dynamics is efficient in some works like controlling temperature in the reaction process,[39] and simulating multiple Ligand unbiniding pathways.[16]

With the development of artificial intelligence, machine learning is also involved in the dynamics of molecular systems.

2.3 Neural networks developments

In the field of artificial intelligence, the concept of machine learning was proposed in the last century. Until today, machine learning approaches have impacted a wide range of industries, including computer science, engineering, mathematics, physics, neuroscience, and cognitive science.[40] It is a method that makes good use of the computing power of machines that cannot be achieved by humans to construct a method to learn from data, and constantly adjust itself, and finally complete the specified task.[41] A specific task can be implemented by an algorithm, and all the machine needs to do is to convert a sequence of input instructions into a sequence of outputs through the algorithm. Different algorithms can achieve different effects, such as accuracy, function, but also require different computing memory and time.[40] What machine learning develops for, is exploring the most efficient and precise algorithm to solve a task from the specific field. According to inputs and feedback of different tasks, the machine learning has been divided into several broad categories, supervised learning, unsupervised learning,

semi-supervised learning and reinforcement learning. The concerned point of this project is supervised learning algorithm, which constructs a mathematical model with labeled inputs and desired outputs.[42]

Artificial neural network (ANN or NN) is a biological neural network stimulated by animal brains, which can be used in the supervised learning task.[43] Born from David Rumelhart's work in 1986, recurrent neural network (RNN) is an important type of artificial neural network that has connections between units with the function of process time series.[44] Multi layer Perceptron (MLP), as a classical neural network, have three or more layers to transmit information forward (see figure 2.3).

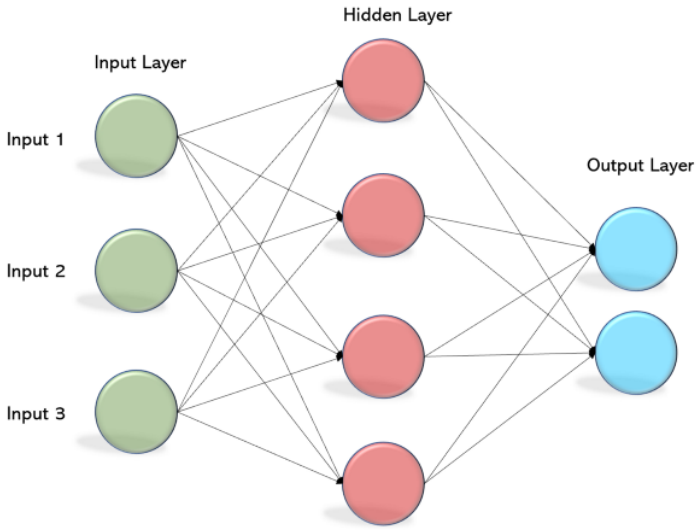


Figure 2.3: MLP structure

In addition to forward connections, there are self-connections in units themselves, or connections among units in the previous layer. The recurrency works as short-term memory, causing the network to remember what happened in the past.[40] However, there is a main limitation about gradient descent in standard RNN architectures, gradient vanishing or exploding exponentially with the size of the long time gap between important events.[45][46]

There are some remedies proposed to deal with the problem in gradient descent. Because learning long time dependency have a strong relationship with the weight search optimisation method, a representative method which did not use weights was came up by Angeline and his group in 1994.[47] The subtlety of this approach is initializing the network weights randomly until causing network can classifies all training sequential data. However, the deficiencies are so obvious that guessing is inefficient or even impossible on a problem with a mass of data.[46] The second approach is aimed to increase the weights during the transmission, Yoshua Bengio etc., proposed a way combined time-weighted Pseudo-Newton optimization and discrete error

propagation.[48][49] Compared with several standard optimisation algorithm, the testing outcome on the tasks where the time gap of the input or output dependencies could be manipulated is satisfactory. Nevertheless, the capacity of reconciling learning with storing real value data remained to be solved. Operating on the higher levels is the third way, Y Bengio and P Frasconi used a EM method for objective propagation, which achieves good results on discrete number of states but has cons on continuous problems.[50][49] In addition, some new network structure has been proposed, C Giles innovatively used higher order recurrent networks.[51] By learning to utilize a stack memory from external, the model understood uncomplicated context-free syntax effectually, which was aimed to increase error flow but the case of error vanishing was scarcely be solved.[51][49] S Hochreiter and J Schmidhuber found Long Short-Term Memory network (LSTM).[52] So far, LSTM has formed a significantly innovative structure (see Figure2.4).

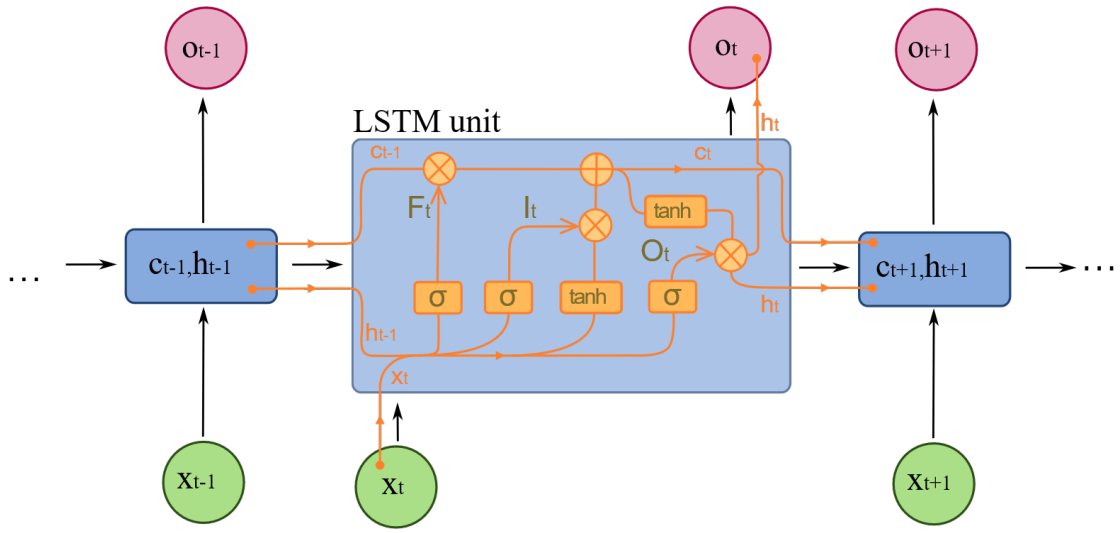


Figure 2.4: LSTM Unit.[53]

In the unit of LSTM, there are three gates and a cell state controlling the flow of information. The first one is the forget gate which gets information of current input X_t and last hidden state h_{t-1}, decides if them should be forgotten through an activation Sigmoid function, and get F_t. The second one is input gate, and the purpose of the gate is putting the X_t and h_{t-1} into the activation function and the $tanh$ function, and multiplying two results. The next step is cell state calculation, which allows that last cell state C_{t-1} times F_t to get the a number. If the number is close to 0, it would abandon the result from input gate, indicating "forget". By contrast, if the number is close to 1, the result from input gate would be retained. After getting the current state c_t, the final output gate would transmit c_t to the next unit. And the output gate passes the current hidden unit to next unit by multiplying two results, where one is from X_t and h_{t-1} through an activation function, and another one is from c_t through a $tanh$ function. In the algorithm of backward propagation, to finding the gradient of a node, the model first find the output node of the node, then calculate the gradient of all output nodes and multiply the gradient of the output

node to the node, and finally add the gradient to get the node's gradient.

Although, LSTM had some pros in bridging capability in long time lags, coping with abundant state number, generalizing greatly when positions are widely separated, and using BPTT (back-propagation through time) locally in time and space respectively, it still suffered from analogous "strongly delayed XOR problems", almost 9 times weights calculation, receiving the entire input string simultaneously, determining inaccurately discrete time steps.[52] Until now it turns out that LSTM has considerable significance in the field of time series prediction,[54] speech recognition,[55] drug design,[56] and etc.

After the creation of LTSM, some alternatives of it have been raised, such as bidirectional long short-term memory (BLSTM),[57] gated recurrent units (GRU),[58] and transformer.[59]

2.4 Machine learning for dynamics on molecular systems

In past decade, a lot of researches in the intersection field of molecular systems and machine learning have emerged. In 2011, Igor Chikalov and his teammates introduced a method for modeling H-bond stability in proteins through binary regression tree.[60] The binary regression tree model is obviously better than the model based on H-bond energy alone. Besides, they proposed some ways to improve the research performance, such as averaging predictor values, using Gradient Boosting Trees,[61] refining the notion of stability.

Some deep learning techniques such as neural networks (NNs) appeared in the molecular systems. In 2016, Bin Jiang got a less RMSE result on predicting the reaction trajectories, by using the neural networks combined with permutation invariant polynomial (PIP-NN) method, than the result only based on PIP.[62] But it was hard to apply the PIP-NN method on larger molecular systems, which have more than 10 atoms. Therefore, he recommended some methods, like differentiating the atoms with less exchanging possibility, or using Gaussian Process to fit potential energy surface (PES).[63] In the same year, Florian Häse applied the multi-layer perceptrons (MLP) technique to quantum/molecular mechanics (QM/MM), and it decreased the calculated amount of the exciton dynamics of large photosynthetic complexes, which saved lots of computational costs.[64] In addition, this method could also be extended to research with other dynamics systems, such as exploring small changes based on exciton dynamics.In Oliver Fleetwood's work (2020), several ML (like MLP) methods were used to explore important features in MD.[65] With this work, the effect of binding of ligand and GPCR, was originally analysed. Compared with MD, the machine learning method are less computationally expensive. Finally, they pointed out that machine learning is possible to demystify more complicated simulations.

As a technique with the ability to process long time information, LSTM has been applied on molecular system simulations. In protein dynamics produced by physics-based simulations,

Chitrak Gupta and etc., in 2020, showed a good forecast ability of LSTM for the data with a high correlation among the recent past structures.[66] In the experiment, non-equilibrium protein dynamics resulted in a great learning for LSTM, whereas, protein at thermal equilibrium required longer history size to keep the accuracy. They gave a summary that LSTMs is an effective approach to simulate non-equilibrium protein dynamics which happens in almost all of the biologically relevant actions. In 2020, Sun-Ting Tsai and his team successfully proved the neural networks models (like lstm) can learning the process of conformational transitions in alanine dipeptide, which are extracting correct Boltzmann statistics and duplicating kinetics across a continuous timescales.[17] And also they applied a embedding layer on training process to find some nontrivial relationships among kinetic distances. The work of Tsai could be a significant step to explore the dynamics in chemistry, biology and physics, based on RNNs.

2.5 Conclusion

With the development of machine learning over the years, neural networks, especially LSTM, have driven the dynamics of molecular systems. We want to train models with a set of CVs based on Langevin dynamical systems and to predict the result of trajectory. In addition, the contribution of features in the training would be detected by MLTSA and LIME. However, due to the dimension of features, we can not simply add an embedding layer into the model, like what Tsai did. We hope our method would be applied on MD systems as well as other dynamical systems.

Chapter 3 **Methodology**

In this chapter, the one-dimensional data generator, two-dimensional data generator, structures of models, and Machine Learning Transition State Analysis (MLTSA) and Local interpretable model-agnostic explanations (LIME) would be elaborated. In one-dimensional data generator, there is the method of code implementation. In two-dimensional data generator, how the spiral data are generated would be stated. The final section would explain how the ML method is implemented in detail, and the evaluating way for models.

The targets of the method is predicting the trajectories result of potentials, and discovering significance of different features in ligand unbinding, in a coding way.

3.1 1D Data generator

The data generator are designed on Numpy and Scikit-learn packages, and dynamic mechanics is based on Langevin dynamics.

In the simulation, we generate trajectories by 2 potentials. One is the single well (SW), and another one is double well which are considered as Noise. First we generate a series of potentials, and choose one among all the DW potentials. The result of the selected DW would be considered as the result of this trajectory. And the results of trajectories are defined as "IN" and "OUT", which are position state of ligand-bound and ligand-unbound respectively.[12] To mix potentials, we would randomly pick two from all generated potentials and mix them by the following formula 3.1.

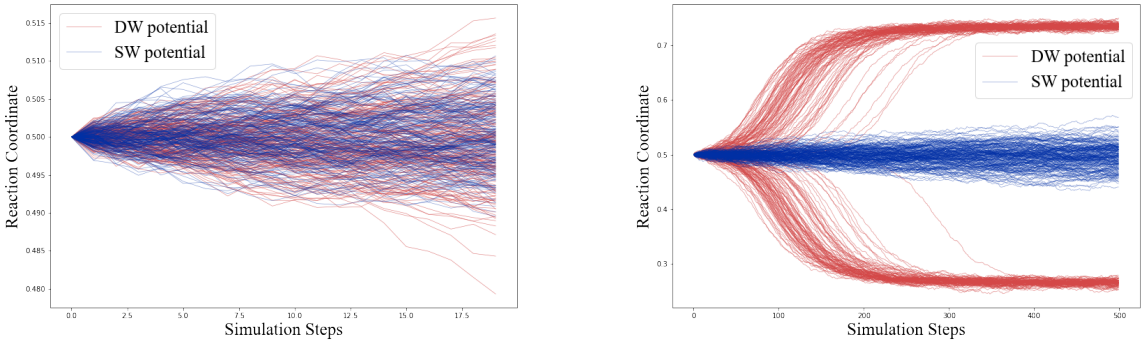
To explore the mechanics of unbinding process, we define the features through a linear equation,

$$y_{feature} = \alpha y_1 + (1-\alpha) y_2 \tag{3.1}$$

, where y is the coordinate of potentials, α is the correlation coefficient to the DW, between 0 and 1.

Finally, no matter which two potentials are mixed, we use the picked DW potential as the crucial one to determine the label for this trajectory. And we record the correlation coefficient.

The program can generate total number of potentials (DW and SW), number of DW potentials, based on user input. The input of time could control amount of the integration step time of the trajectories. There is also a input to operate the number of simulations. The below figures 3.1(a) and 3.1(b), are trajectories produced with 25 total potentials (SW and DW), 5 DW potentials and 200 simulations, running 20, 80, 200 and 500 integration steps respectively.



(a) (b)

Figure 3.1: 1D data with step 20 (a) and step 500 (b)

From the figures 3.1(a) and 3.1(b), the abscissa and ordinate represents the reaction coodinates and duration of simulation time, and they illustrate the trajectories of two potentials moving from the transition state.

Each simulation would emerge different random correlation coefficients α to manipulate feature, with a input of feature number. And the feature coefficient is fixed in each simulation. The figure 3.2 demonstrates two trajectories with different features in two simulations.

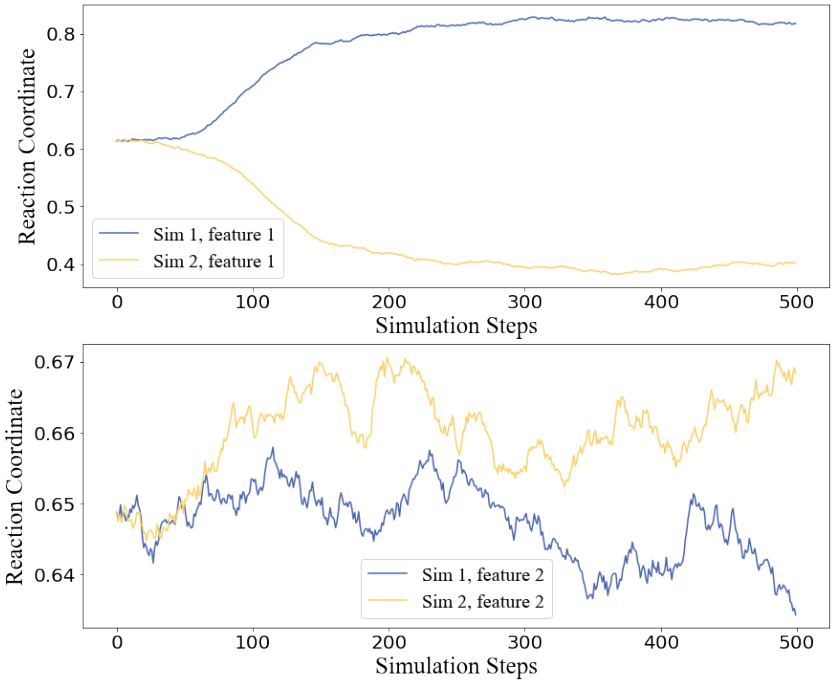
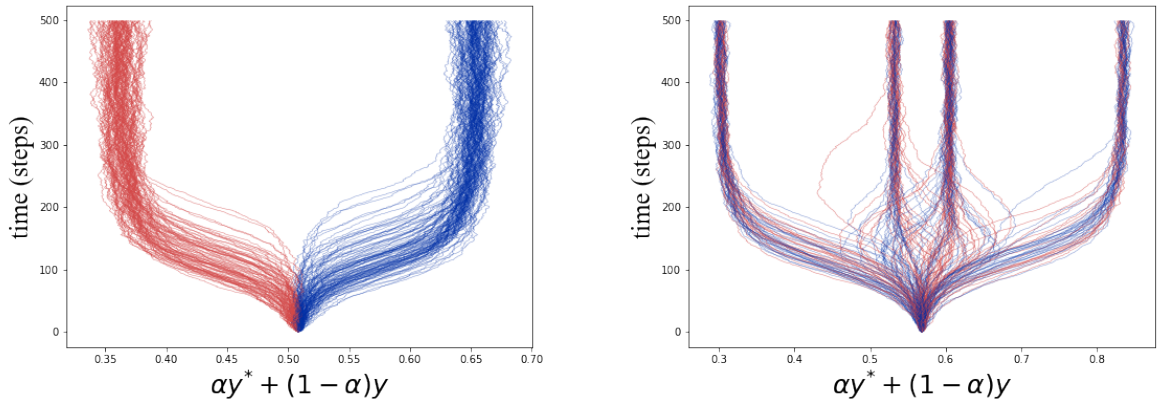


Figure 3.2: Features in simulations. The sub-figure above illustrates the first feature in the first and second simulations. The sub-figure below shows the second feature in the first and second simulations.

In the figure 3.2, two combined potential features move as integration step increasing. In two simulations, the same features present distinct trajectories. In one simulation, the two features show large differences, which is the main focus of the project.

The following figures show how the feature created by different α in a more visualized way.



(a) (b)

Figure 3.3: 1D data feature. (a) and (b) present all the simulations of feature 60 and 63.

From the figure 3.3(a) and3.3(b), some different features illustrate different shape. Trajectories of a part of features are separated clearly, and some are mixed up, which could be analyzed in a hard level with human eyes. So a machine learning method is applied to classify the "IN" and "out".

Finally, the data would be transferred to the shape of (number of simulations, time frame, number of features), in Numpy arrary formation.

And we suppose that the features with high correlation values have a good significance in the unbinding simulation.

3.2 2D data generator

The 2D data generation was taken a novel approach which is not linear combination, based on the Langevin dynamics simulation. Each simulation generates a series of coordinates of potentials moving from transition state. In this section, the process of data generation of all spiral shape would be explained.

At first, one point would start from the center on a potential surface which is based on the potential function. There are two force, where one towards the gradient of the potential surface, and another force causes the point to follow the Brownian motion. Finally, this point will form an irregular trajectory roughly along the gradient of the potential energy surface. And, the we map the position to a plane, collect the x and y coordinates and record the final potential. After generating hundreds of trajectory, we got 2D data and converted it into the shape of (number of

simulations, running time).

With the 2D data generator, there are some shapes of spiral data, and we choose the data with 2 branch to explain in the project. The projection of the spiral data on the plane where the x- and y-axes lie is similar to the shape of the DW potential. And on the figure 3.4, the position with high distribution will be represented by a dark color. Conversely, when the potentials at one point are small, it will be illustrate by light colors.

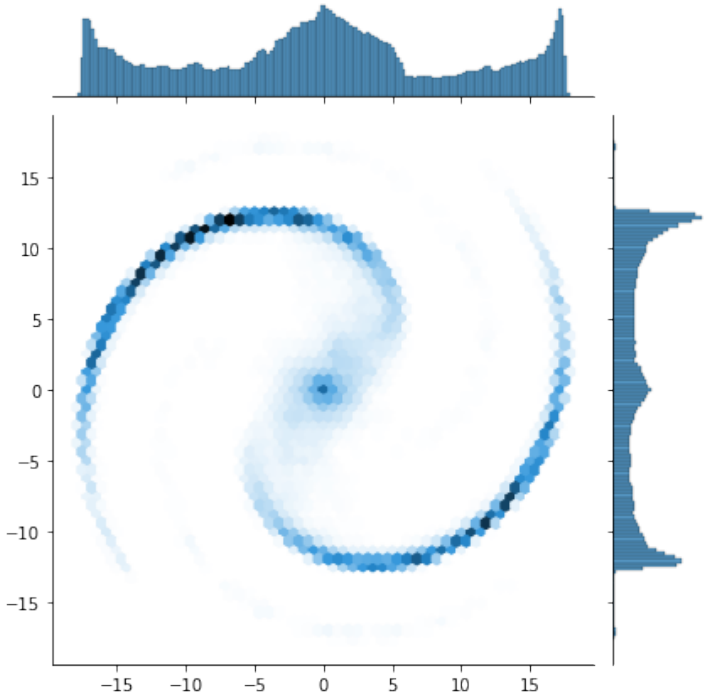
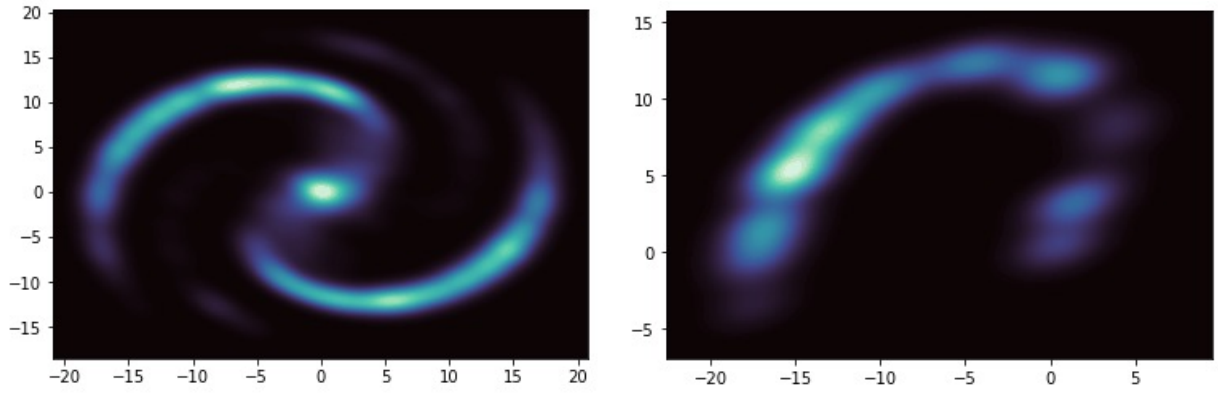


Figure 3.4: Distribution of Spiral 2D data

The figure 3.4 presents the spiral data with two branches ,and there are DW distributions projections on both top and right side. And we could notice that the shape of the distribution on the plane is like a lateral "S".

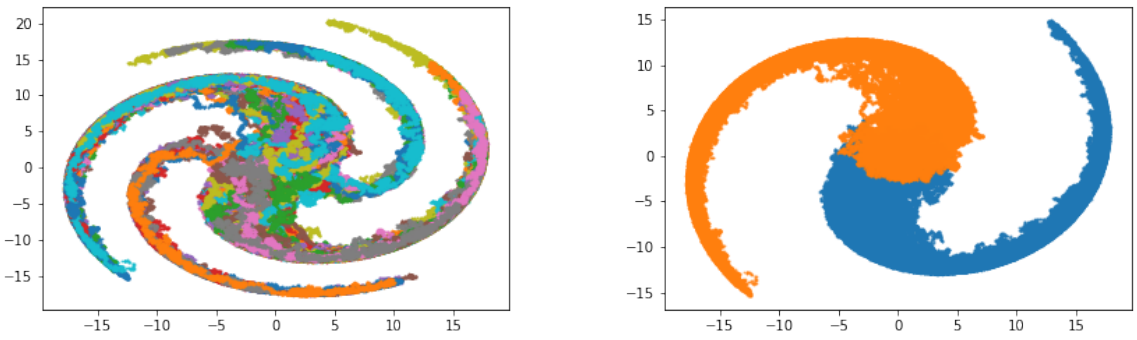
The following images 3.5(a) expresses the distribution of potentials in a more visual way that is similar as a topographic map. There are black on position of less distribution, and bright on position of more distribution. And the images 3.5(b) demonstrate the distribution of one simulation.



(a) (b)

Figure 3.5: Spiral data. There are the data distribution of all simulations (a) and one simulation (b).

The code is also implemented by Numpy package. In the beginning, the program can produce distribution with input of total number of DW potentials. And the input of time could manipulate the number of the integration steps. The number of simulations is also the necessary input. After the input has been process, the program produce a raw data with some amount of noise (see figure 3.6(a)).Then the is a function to filter the noise data, and we get the clean data(see figure 3.6(b)).

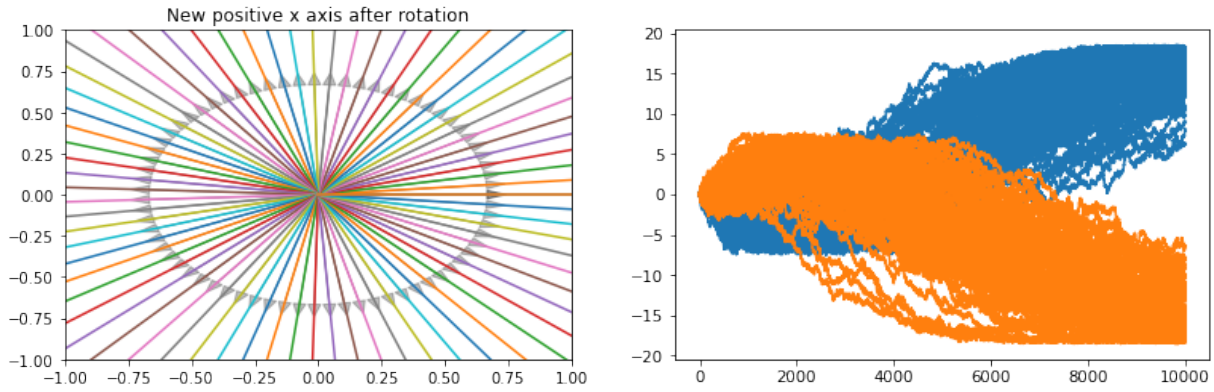


(a) (b)

Figure 3.6: Raw data (a) and clearn data (b)

In order to create features of 2D data, we create a series of uniform spaced coefficients, between 0 and 2π (unit: rad). These coefficients represent the angle of rotation counterclockwise, starting from the horizontal line to the right. The figure 3.7(a) shows 72 features in the a plane. The

figure 3.7(b) is a view of one feature from a cross-sectional perspective. The X-axis represents the time scale.



(a) (b)

Figure 3.7: Spiral data feature. Figure (a) shows all the figures we took. Figure (b) one of the features viewed from a cross-sectional perspective.

Finally, the data would be processed as the shape of (number of simulations, time frame, number of features), in Numpy arrary formation.

3.3 Models training

3.3.1 Training architecture

Magd Badaoui and Pedro J. Buigues developed GBDT and MLP method to distinguish the "IN" and "OUT" with given labels.[12] In this project, the output results are discriminated with a LSTM architecture. LSTM could retain information from a long time in the past and apply it to a new point in time.

The whole machine learning architectures are build with tensorflow2, Numpy package. And we use Keras to build the blocks of vanilla LSTM. The LSTM need a input data with the structure of (times of simulation, time frame, number of features). Thus, we generate 200 MD simulations data with 1000 integration steps, create 72 features for it, and change it to the correct shape as Numpy array. 72 features is to keep the same level of 2D data generation, and result in a relatively short calculation time. Correspondingly, the labels of each simulation are given by the 1D data generator.

Then in order to predict results in different time nodes, we decided to training models with

data from different time frame. In order to automatically correspond to data with different time frames, new models need to be constantly created. The flow of training are displayed below.

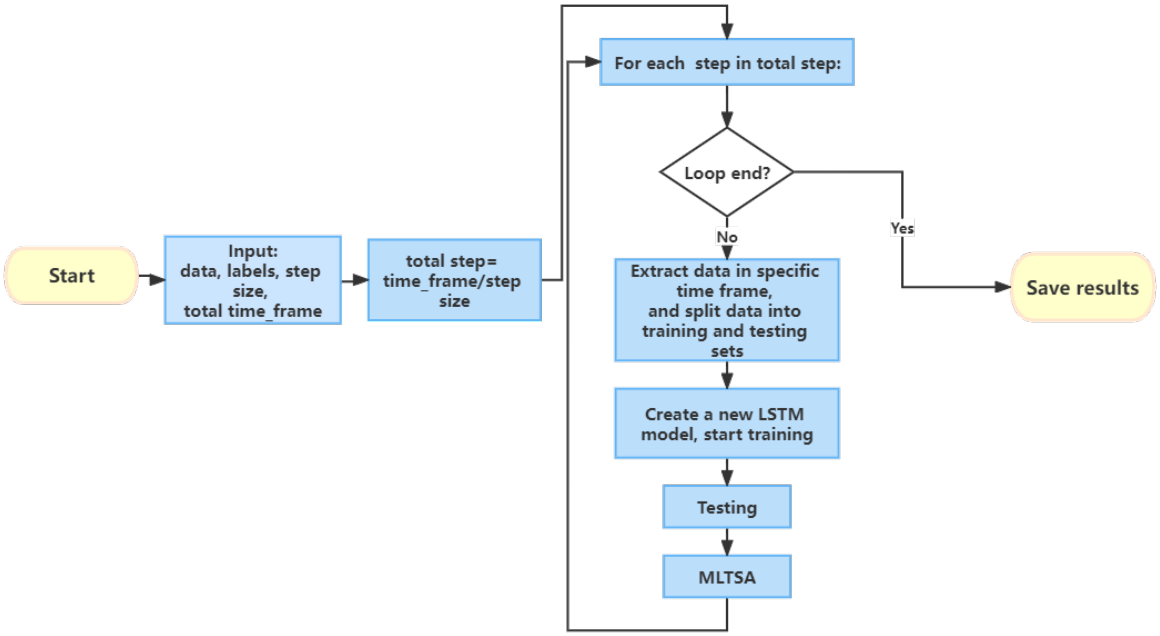


Figure 3.8: Training flow chart

According to the flow chart, it would be summarized as three parts. In the first part, the program will receive the data of the correct shape, the corresponding labels, the total running time, and the span of each step. Then, with formula $timeframe/stepsize$ to get the total number of steps, which is also the number of models should be created.

The second part is create model based on a for loop. First of all, the data would be extracted to specific data frame, such as 0 to 20 and 0 to 40. Afterwards, the data with small time frame would be individed into training set and test set. So before everything starts, we run enough simulations (200 times) to guarantee enough training data. Next, the structure of the model will be automatically generated based on the shape of the training set, and the labels would transfer to (number of simulations, types of label) shape through the one-hot encoder. And training will begin.

The third part of program would evaluate the accuracy and loss of the model with the testing set. The final step is the MLTSA, where the program will remove each feature one by one, then recalculate the accuracy and compare it to the average accuracy. After all the evaluation procedure finished, there is a dictionary to store the value (one terminology of dictionary) of results, labeling the key (one terminology of dictionary) of current step. Due to the particularity

of the machine learning model, it cannot be stored in a dictionary. So each model would be saved as in a specific folder.

Then program move to the next circulation, until all the step have been run. So that, the output of the program would be a series of folder containing a model and a dictionary containing all the evaluation result.

After all, we generate several programs based on Matplotlib package to view the evaluating result. All the plots would be presented in Chapter 4. In addition, the "for loop" training pattern could also be applied on other model, such MLP, stacked LSTM and bi-directional LSTM.

3.3.2 Models

Vanilla LSTM:

Inside the LSTM model, the first layer is one input LSTM layer, the second layer is a fully connected network layer, and the last layer is an output layer with 2 nodes where the activation function is sigmoid. And the dropout ratio is set to 0.1, to aviod overfitting. In addition, categorical crossentropythe loss function, Adam optimizer, accuracy metrics are chosen for the model. Before the training, validation set proportion is 0.2, maximum epochs is 1000 and an earlystopping with threshold value of 1e-8 and patience times of 5.

Stacked LSTM:

Inside the Stacked LSTM model, the first five layer is a five stacked LSTM layer, the sixth layer is a LSTM layer, the seventh layer is a fully connected network layer, and the last layer is an output layer with 2 nodes where the activation function is sigmoid. And the dropout ratio is set to 0.1, to aviod overfitting. In addition, categorical crossentropythe loss function, Adam optimizer, accuracy metrics are chosen for the model. Before the training, validation set proportion is 0.2, maximum epochs is 1000 and an earlystopping with threshold value of 1e-8 and patience times of 5.

Biodirection LSTM:

Inside the Biodirection LSTM model, the first layer is one bidirectional LSTM layer, the second layer is a fully connected network layer, and the last layer is an output layer with 2 nodes where the activation function is sigmoid. And the dropout ratio is set to 0.1, to aviod overfitting. In addition, categorical crossentropythe loss function, Adam optimizer, accuracy metrics are

chosen for the model. Before the training, validation set proportion is 0.2, maximum epochs is 1000 and an earlystopping with threshold value of 1e-8 and patience times of 5.

Multi-Layer Perceptron (MLP):

Inside the MLP model, the first layer is one input layer, the second and the third layer are fully connected network layers, and the last layer is an output layer with 2 nodes where the activation function is softmax. And the dropout ratio is set to 0.1, to aviod overfitting. In addition, categorical crossentropythe loss function, Adam optimizer, accuracy metrics are chosen for the model. Before the training, validation set proportion is 0.2, maximum epochs is 1000 and an earlystopping with threshold value of 1e-8 and patience times of 5.

3.3.3 Evaluation method

Accuracy:

Accuracy is a common metric used to describe how well a model classifies all classes, which is determined as the ratio of the number of correct predictions to the total number of predictions. In the dichotomous problem, the formula is

$$Accuracy = \frac{True_{positive} + True_{negative}}{True_{positive} + True_{negative} + False_{positive} + False_{negative}} \tag{3.2}$$

.

In our case, the accuracy of each model (in current time step) would be stored, and the mean value of them would be calculated as the accuracy in this time step. The final plot would be a series of accuracy corresponding to a series of time steps.

Machine Learning Transition State Analysis (MLSTA):

MLTSA is a code tool to analyze the importance of low-dimensional features, for complicated molecular dynamical processes, especially for unbinding process of ligand and protein.[12] The input of MLTSA is the whole data. Then the program calculate gloabl eans, excludes feature one by one in the process of re-calculating the accuracy, and calculates the accuracy (number of features) times in total. Finally, MLTSA outputs a series of the accuracy (number of features), which could be used to plot a accuracy drop figure, like 3.9. The x-axis indicates the different features, and the y-axis represent the accuracy after excluding current feature. Therefore, the importance of feature can be directly understood with the accuracy below average. In the example, the No.70 feature play a vital role in the unbinding process.

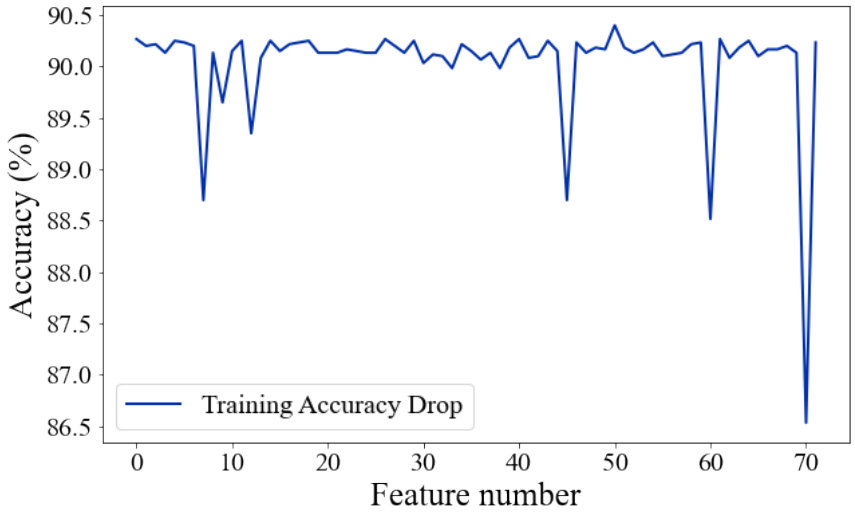


Figure 3.9: Accuracy drop example

Local interpretable model-agnostic explanations (LIME).[67]:

This program is set to explain a machine learning model, which are known as a black box. The approach of the LIME is similar to a new attempt of training but focusing on a local area(see figure3.10). First, the program takes an instance as input, creates some new instances around it, and measures them by how proximate they are to the inputted instance. Next, the program utilizes the ML model to make predictions for these instance, stores them as new training data. The following step is create a interpretable model with the new training data, and focusing on data around the inputted instance. Then we can get the weights at each time step of the trained model to understand all of the features in the inputted instance.

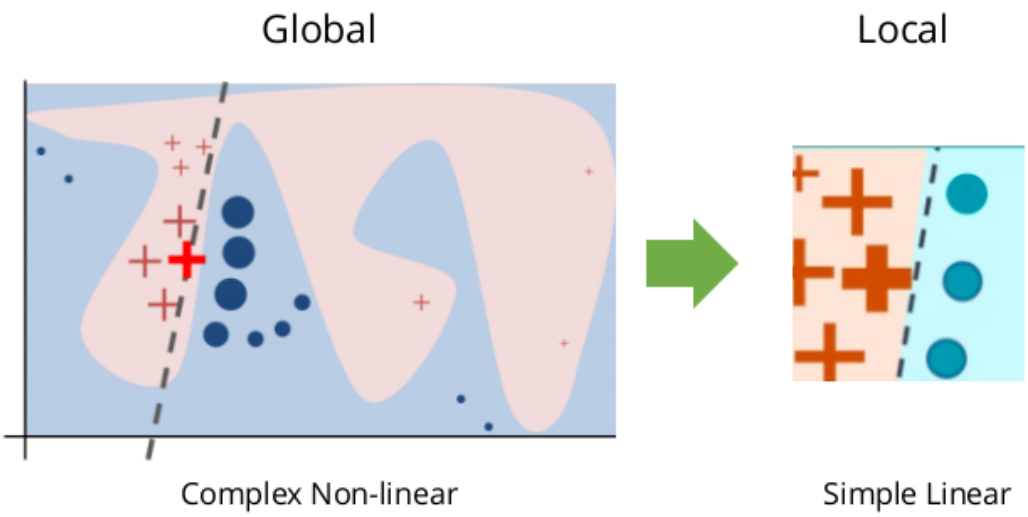


Figure 3.10: LIME explains the model locally.[68]

Because the our model are LSTMs, the RecurrentTabularExplainer class was taken to interpret, which have the same input data shape (number of samples, time steps, number of features) with our models. And we use LIME to explain the instance from our test set. The LIME result would show contributions by each features at each time(feature 1 at time 0, feature 1 at time 1, ... , feature 2 at time 0, feature 2 at time 1, ...).

Chapter 4 **Result and Discussion**

In this project, we completed the training of 1D and 2D classification tasks, and analyzed the important features of the unbinding process. In both 1D and 2D, tensorflow-based, MLP, LSTM, stacked LSTM, and bidirectional LSTM are innovatively used for training classification tasks. The trained models of LSTM and MLP were used to analyze the important features in the decoupling. In 1D, the trained models of LSTM was used to analyze the important features in dynamical unbinding process, and MLP were tried to analyze the same thing. In 2D, only the model of LSTM was applied.

4.1 1D result

4.1.1 "IN"/"OUT" prediction

LSTM accuracy:

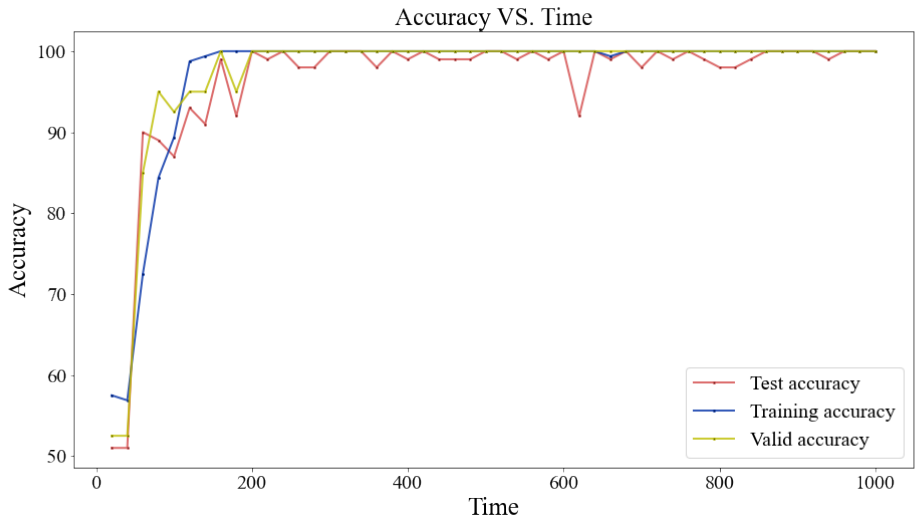


Figure 4.1: Accuracy of lstm model

In the figure 4.1, there are three lines representing the records of LSTM (vanilla) model training, where x-axis is the time point with step 20, and the y-axis represents the accuracy at this time. The training accuracy starts from around 60 in time 20, after a big jump it reaches roughly 100%

accuracy at time 120 and keep the such high accuracy level at a later time. The start point of testing accuracy is little lower than that of training accuracy, which is slightly higher than 50%. Then, the testing accuracy overtakes the training accuracy by around 17% at time 60, and is overtaken again by training accuracy at time 100. In the next time points, the testing accuracy fluctuates between 90% and 100% that are never higher than training accuracy. Nevertheless, there is a large drop at time 620. For the validation accuracy, it follows climbing trend of testing accuracy before time 200, and keep a almost same level with training accuracy after that time.

Before time 200, the testing accuracy and training accuracy are not much different, which are both around 90%, but after that, the training accuracy is almost always maintained at 100%, while the test accuracy cannot be maintained at 100%. From this, we can conclude that the LSTM model have a fairly strong predicting ability in 1D dynamical unbinding process, but is overfitting to a some extent after time 200. The overfitting may result from the "for loop training" pattern lacking hyper parameter adjusting. And this could be advanced by some approaches adjusting hyper parameters in real time, in the near future.

Four models training accuracy:

In the training process for all models, we also monitor the speed of learning. Besides MLP, LSTM has the fastest speed of learning, and stacked LSTM and bidirectional LSTM are top 2 and top 3 fast models repectively. Due to the different input data, MLP shows a rapid speed of learning. But we tried a same input data for MLP, speed of learning gradually decreases from high, and finally become extremely slow due to the huge matrixes calculation. Therefore, the input data of LSTM could be seen as a matrix dimensions reduction way to avoid huge matrixes calculation.

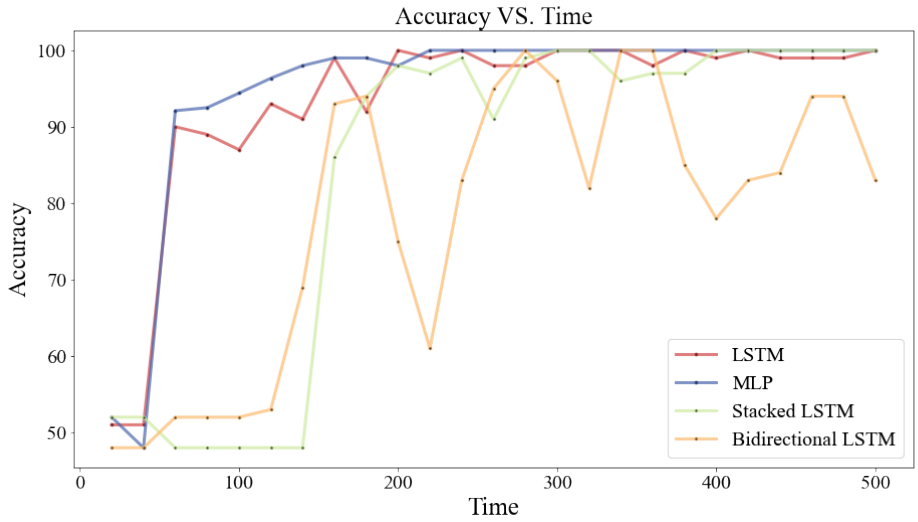


Figure 4.2: Accuracy of 4 models

In the figure 4.2, there are training accuracy polylines of four models, LSTM, MLP, stacked LSTM, and bidirectional LSTM. In the early stage (before time 200), the models with faster training accuracy increment are LSTM and MLP, and they keep a remarkably high accuracy level closing to 100% after that. However, LSTM model show a greater volatility than MLP for the whole time. Other two models, stacked LSTM and bidirectional LSTM, have unsatisfactory prediction accuracy before time 140. After a big jump, they both reach to 90+% accuracy, and then stacked LSTM fluctuates between 90% and 100% and then converges to 100%. Whereas, the bidirectional LSTM is extremely unstable after reaching 90% and it is hard to see a convergence.

From the early time (60), due to the independence between features and time, both LSTM and MLP have shown reliable prediction results, and the prediction accuracy of MLP is slightly higher than that of LSTM. Therefore, both of them could be satisfactory models in 1D prediction. However, there are certain degrees of overfitting (100% predicting accuracy) for both models. For the stacked LSTM model, although the prediction accuracy in the early stage is dissatisfactory, the prediction accuracy in the later stage shows a convergence trend to 100, which can also be used as a suitable predictor. But the bidirectional LSTM could not be seen as a qualified predictor because there is no convergence tendency even at time 500. Due to the limitation of calculating resources, we only trained bidirectional and stacked LSTM from time 0 to time 50, which could be a reason why the bidirectional LSTM does not converge.

Interestingly, all LSTM models perform a periodic fluctuation, which may be caused by the long-term memory function. This exceeded our expectations because in 1D simulations the

features do not change over time, and it could be a important field to explore the process of protein-ligand unbinding for the following studies.

4.1.2 Important feature detection

To analyze the 1D features in the unbinding process, we need to extract features from some important times.

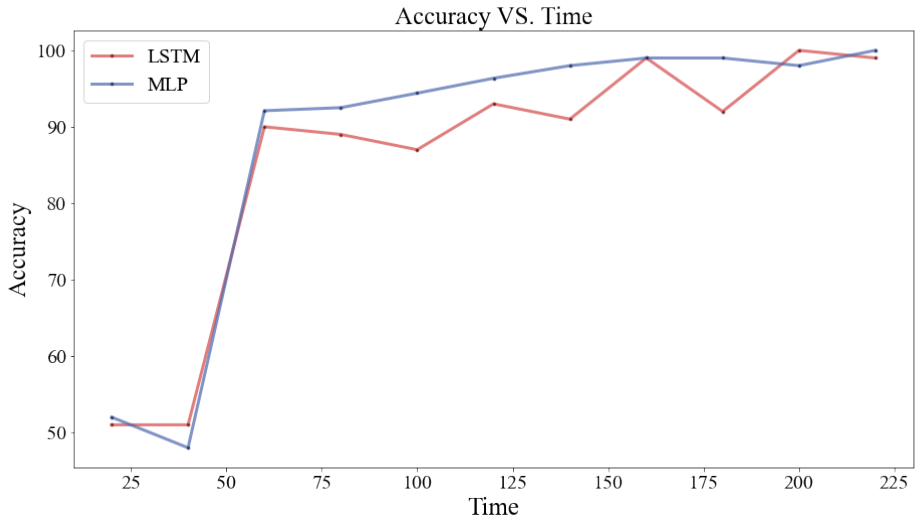
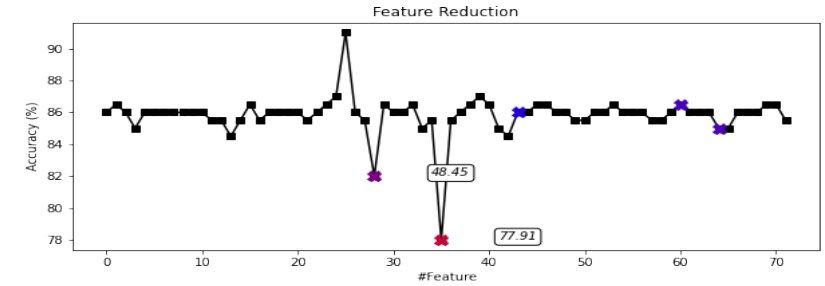


Figure 4.3: Accuracy of LSTM and MLP before time 220

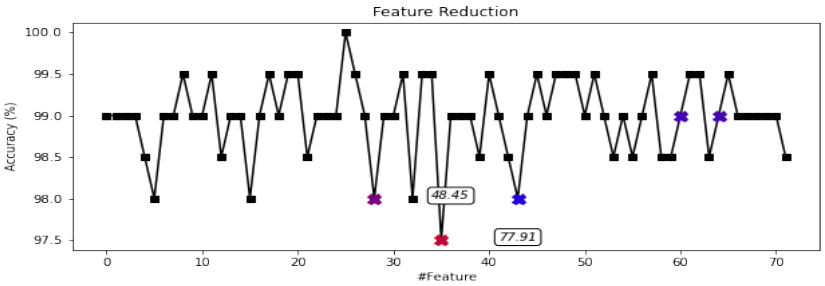
The figure 4.3 is the detailed LSTM and MLP version of figure 4.2 from time 0 to 200. And from that, we pick 3 points at time 60, 160 and 200.

MLTSA for LSTM:

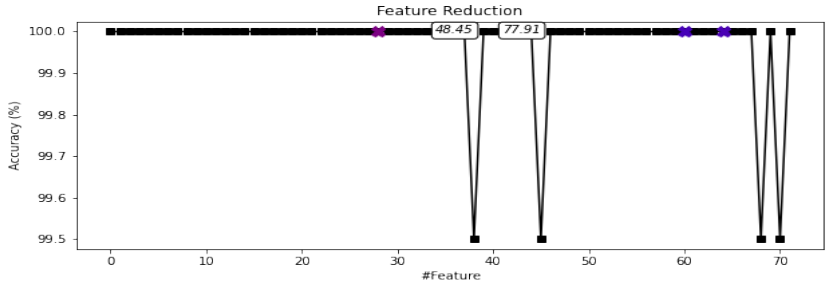
The following figure 4.4(a), 4.4(b) and 4.4(c) are the accuracy drop figure by MLTSA at time 60, time 160 and time 200 respectively, and the figure 4.4(d) is the color figure corresponding to the correlation value to the vital DW potential.



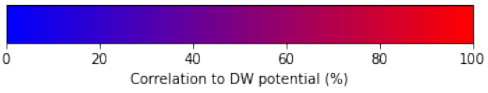
(a)



(b)



(c)



(d)

Figure 4.4: Accuracy drop of LSTM. Sub-figures illustrate the accuracy drop at time 60 (a), time 160 (b) and time 200 (c). The sub-figure (d) is the color of the correlation coefficient.

In the above figures, we show the features with correlation value bigger than 30, and mark them with a color from figure 4.4(d). The there are total 5 features are marked in each figure. At time 60 and 160, the most important are the feature 36 (from 1 to 72) with the correlation value 77.91, whereas it was not the most important feature at time 200. At time 200, there are 4 most

important feature in No.37, No.44, No.67 and No.69, whose correlation value are obviously lower than 77.91.

There are no 100% training accuracy at time 60 and time 160, and the features with high correlation values at both of times shows a satisfactory results about unbinding importance. Nevertheless, features at time 200 do not show that good result, which may caused from the overfitted 100% training accuracy.

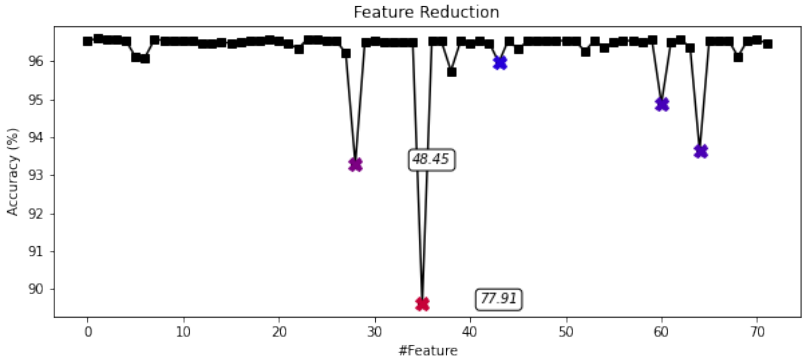


Figure 4.5: Accuracy drop of LSTM through 1000 time

And the figure 4.6 illustrates the accuracy drop through all times by superimposing all the accuracy drop data. It is clear that features have top correlation values have more importance in the unbinding process.

Overall, we could obtain two conclusions. The first one is that model LSTM do prove the hypothesis, the high correlation feature has high importance in the unbinidng simulaiton, and validate the feasibility of MLTSA. The second one is the overfitting data with 100 percent are unbelievable, and should be avoided in the future works.

MLTSA for MLP:

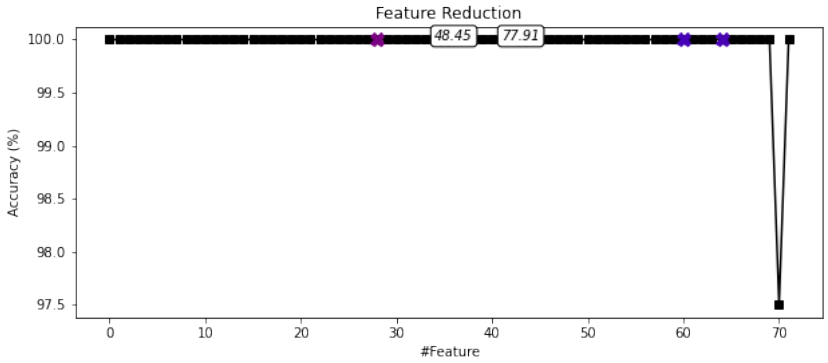


Figure 4.6: Accuracy drop of MLP at time 160

But in the MLTSA results for the MLP model, they are not satisfactory, where the importance of the features do not follow the DW correlation value.

4.2 2D result

4.2.1 "IN"/"OUT" prediction

LSTM accuracy:

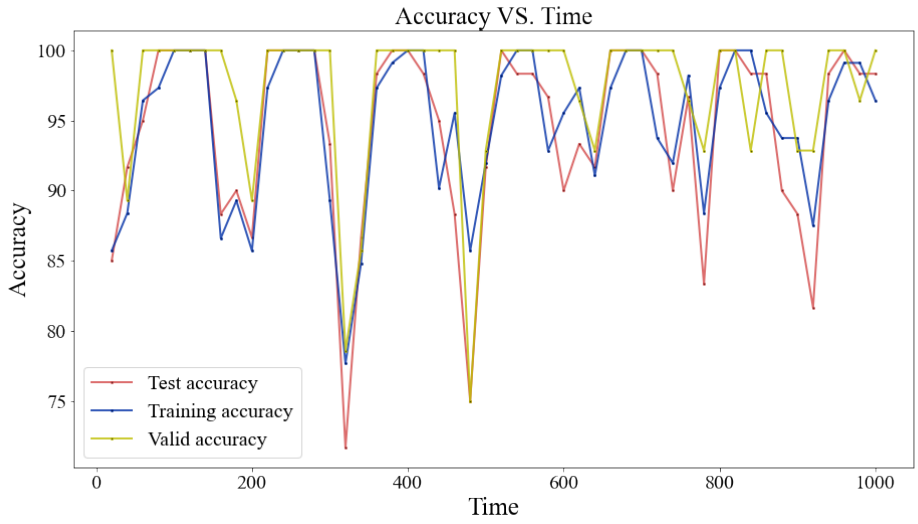


Figure 4.7: Accuracy of LSTM

Figure 4.7 shows the training, validation and testing accuracy of LSTM model from time 0 to 1000. The training accuracy is high at the beginning, reaching around 86, and rise steadily to 100% at time 100. For the testing accuracy, with a begging point similar with training accuracy, it reaches 100% at time 80. After that, accuracy of all three set fluctuating between 70% and 100%. Besides, the accuracy of three sets do not show a convergence trend from time 0 to 1000.

The advantages of LSTM is that it has nice predicting ability before time 300 which is higher than 85%. However, after time 300, the accuracy difference between testing set and training set become larger, and testing accuracy falls below 85% at time 320, and show a big fluctuation. The large accuracy difference may results from the "for loop" training pattern, which lacks capability of adjusting hyper parameters during the training. Therefore, the accuracy of LSTM could be trusted only in early times.

Four models training accuracy:

In 2D simulation, four models similar to that in 1D have been trained, and the speed of learning is similar to that case in 1D.

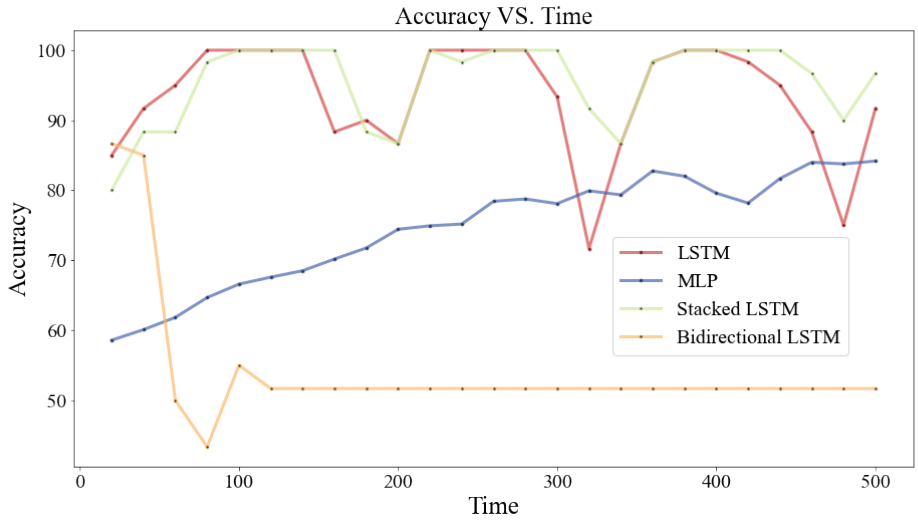


Figure 4.8: Accuracy of four models

There are four accuracy lines before time 500 in figure 4.8. LSTM and stacked LSTM show approximate accuracy results, but stacked LSTM has a more stable fluctuation. For the MLP, its accuracy climbs slowly with a start point below 60% and finally converges around 80%, which are almost always less than lstm accuracy. Besides, the bidirectional LSTM does not have acceptable accuracy results which are slightly higer than 50%, even with the highest accuracy in the beginning.

From the comparison among four models, LSTM and stacked LSTM present the best predicting ability. The reason may be that the long-term memory capabilities of the model are able to capture features that change over time. And MLP may also be unable to make great predictions for 2D simulations due to lack of long-term memory. The bidirectional LSTM shows bad results in both 1D and 2D, which may because the way the data is inputted in both directions is not suitable for the simulation of protein-ligand unbinding.

Similar to 1D, accuracy of stacked LSTM and LSTM perform periodic fluctuations between 100% and 70% through time. The reason for this phenomenon may be caused by that the angle of the new data is changed compared to the angle of the previous data, and the model needs enough data to adapt to the change of angle.

4.2.2 Important features detection

We chose the LSTM model to analyze important features during the 2D simulation. As the evaluation method is still under development, we just present the important features with MLTSA and LIME.

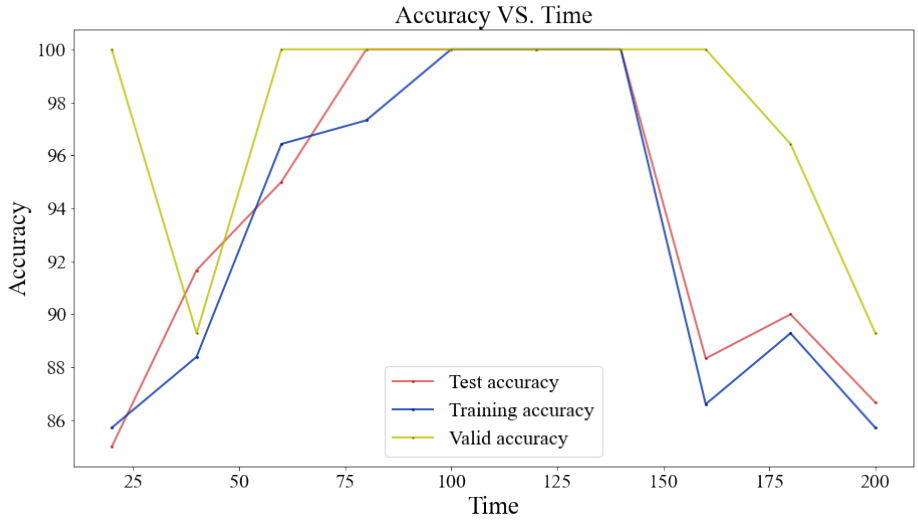


Figure 4.9: Accuracy of LSTM until 200

The figure 4.9 is the detailed version of figure 4.7 from time 0 to 200. And from that, we pick 3 points at time 60, 160 and 200, Which are non-100% accuracy, 100% accuracy and the accuracy fell down from 100%.

MLTSA for LSTM:

In the following figures, the top 3 important features are shown on the left, and the data (clean data) location with "In" and "OUT" labels is on the right. More than one feature have the same order because they have the same importance at the current time.

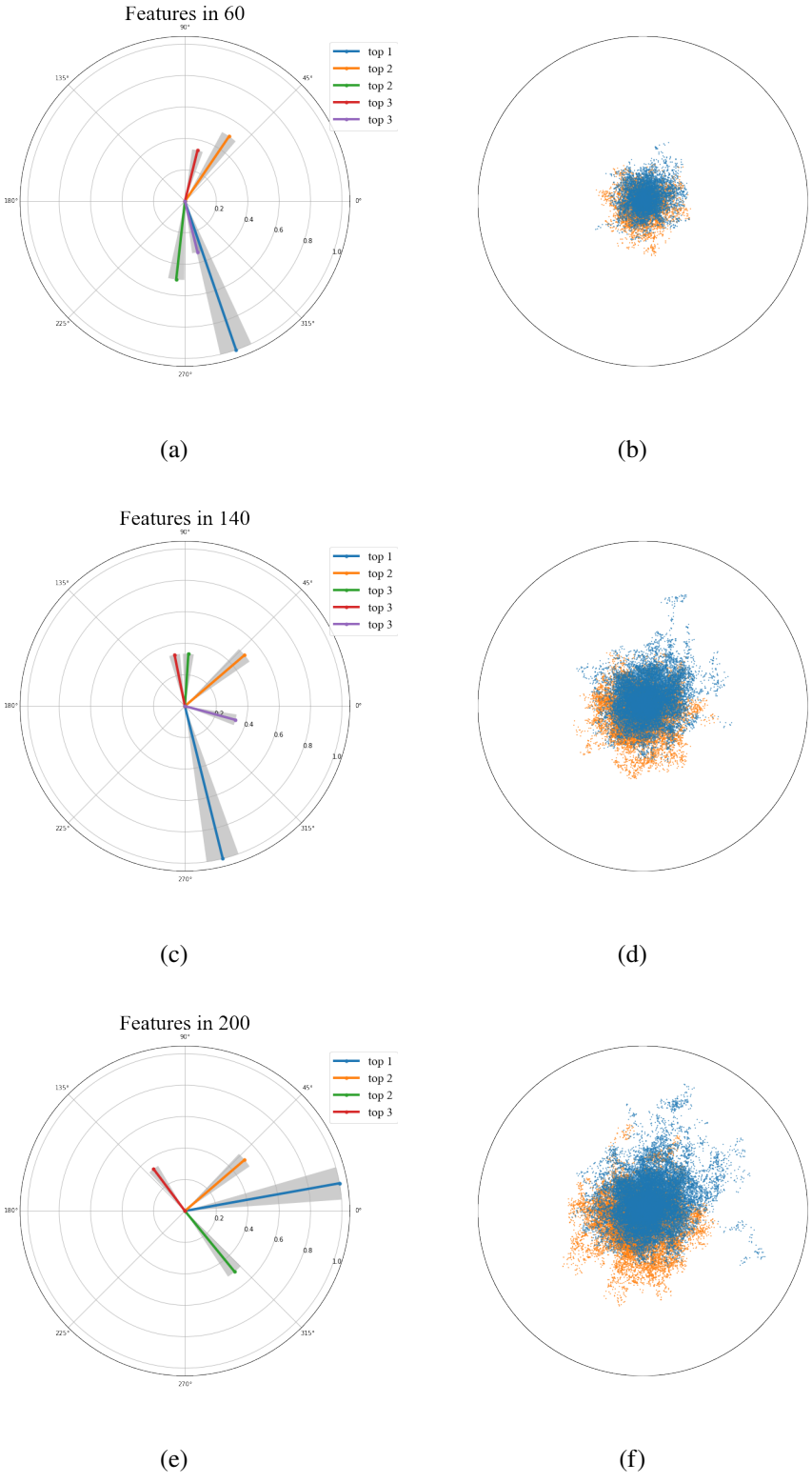


Figure 4.10: Important spiral data feature by MLTSA. The left and right sub-figures are important features and data distributions at at time 60, 140 and 200.

The figures 4.10(b) 4.10(d) and 4.10(f) illustrate the data location on a plane at time 60, 140 and 200 respectively. The figure 4.10(a) at time 60 and the figure 4.10(c) at time 140 illustrate similar top 1 features with around 290 degrees, where there are some little angle changes. For the figure 4.10(e), the degree changes from lower right corner to upper right corner with 10 degree. In addition, they all have a top 2 feature around 45 degree.

With the MLSTA, we successfully found the important features in the unbinding simulation process, and the most important feature moves as time. For the top 1 feature at time 200, it may be caused by the underftting without the enough new data when the direction where the new data is generated is deviated from the direction where the previous data was generated. And the "for loop" training pattern may also be a part of the reason. This problem could be mitigated with more data generated and adjusting hyper parameters in real time.

LIME for LSTM:

There are important features analysis dendrogram by the LIME, where the feature are divided into $n_{currenttime}$ features, and each subscript corresponds to a certain time (start from 0).

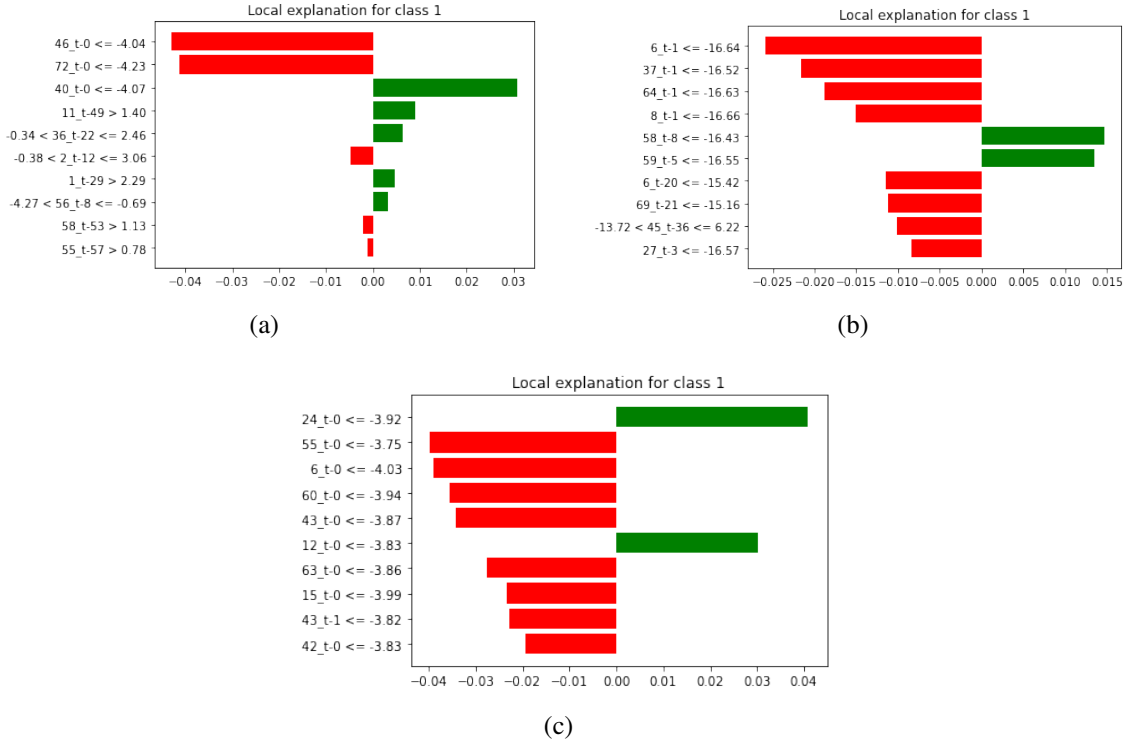


Figure 4.11: Important spiral data feature by LIME. The sub-figures present important features ranking at time 60 (a), time 140 (b) and time 200 (c)

The x-axis is the detection contribution from the feature, and some most important features are presented on the y-axis. In addition, the red and green color indicate which result the feature contributes to. In this project, due to the limitation of the calculation resources, we only choose the first sample from the test set, and the label corresponds to the red color.

The top 1 features in figure 4.11(a) 4.11(b) 4.11(c) are 46 (time 0), 6 (time 1) and 24 (time 0), which are totally different. However, if we focus on all the features shown on the figures, there are many contribution between features 50 and 70, which are similar to the results with MLTSA.

In summary, we did a good job on finding the important features at each time point which could be a new area to explore the features in the unbinding simulation. Besides, there is some work to be finished to present a better comparison with MLTSA. Superposition of the contributions from all the same features could show the importance like the way of MLTSA. Using more samples to get a mean value could be a better detector, whereas it would take equivalent to dozens of times the MLSTA calculation.

Chapter 5 **Conclusion**

More recently, the discovery of effective new drugs depends on the length of time the drug and target are bound. The combination/separation of drug and target is defined in the field of biochemistry as ligand binding/unbinding. With the development of machine learning technology, they are used in various dynamic simulations to observe CV based on molecular systems, such as MD, langevin dynamics. Several models of neural networks have also been shown to have reliable predictive capability in dynamic simulations.

In our work, we successfully generate 1D and 2D trajectories analoguous to the ligand unbinding, by linear combination method and two-dimensional combination method, and convert them into the data shape required by MLP and LSTM. In 1D, the prediction results of both the MLP model and the LSTM model are quite satisfactory, as the generated features do not change over time. And all LSTM models perform some degree of periodic fluctuation on accuracy. In the detection of important features based on the LSTM model, MLTSA successfully demonstrated the conclusion that high DW correlation coefficients causes important features, and showed some wrong features at poorly trained time. Our models and tools can be adapted not only for ligand unbinding simulation analysis, but also for physics-based dynamics simulations broadly. In 2D, LSTM and stacked LSTM models show great predictive ability in early data and periodic fluctuations on later accuracy. Compared with these two models, MLP does not perform well in 2D prediction results. In addition, both MLTSA and LIME successfully demonstrated the analysis results for important features. Through visualization methods, MLTSA shows how important features change as new data is generated. The results of LIME are more detailed, showing contributions of all features at all times. From the result before time 500, we could guess that LSTM performed better than MLP at a later time. Moreover, the detection results of MLTSA and LIME have certain similarities.

Also, there are some limitations. First, in 1D and 2D, the "for loop" training pattern lacks hyper parameters tuning for new models at later times, which leads to underfitting and overfitting to some extent. The training pattern can be improved in some ways, such as adding cross-validation, or pre-training first and then fine tuning. Second, due to the insufficient speed of learning of MLP, the time frame of the input data of MLP is different with that of LSTM, which leads to the inability to accurately judge the difference between the results of MLP and LSTM. This can be done by changing the training environment, such as running on a server. Third, in 2D, tools for validation of MLTSA and LIME are demanded. This can be verified using the JaccardIndex method.

In addition, there are a few things that could be included in future works. The first is to improve and analyze more comprehensive of MLTSA for MLPs in 1D. The second is to explore periodic fluctuations in LSTM training accuracy in 1D. The third is to create visualization tools for lime to compare MLTSA results. Next, investigating the poor performance of bidirectional LSTM can be a meaningful study. Finally, adding an embedding layer to find some relationships between times would be a challenging work.

Reference

[1] Deborah Leckband and Jacob Israelachvili. Intermolecular forces in biology. *Quarterly reviews of biophysics*, 34(2):105–267, 2001.

[2] TW Graham Solomons and Craig B Fryhle. *Organic chemistry*. John Wiley & Sons, 2008.

[3] G Michael Blackburn, Anita Datta, Hazel Denham, and Paul Wentworth. Catalytic antibodies. In *Advances in Physical Organic Chemistry*, volume 31, pages 249–392. Elsevier, 1999.

[4] Christopher D Fu, Luiz FL Oliveira, and Jim Pfaendtner. Assessing generic collective variables for determining reaction rates in metadynamics simulations. *Journal of Chemical Theory and Computation*, 13(3):968–973, 2017.

[5] Ru Zhang and Xin Xie. Tools for gpcr drug discovery. *Acta Pharmacologica Sinica*, 33(3):372–384, 2012.

[6] Alexander S Hauser, Misty M Attwood, Mathias Rask-Andersen, Helgi B Schiöth, and David E Gloriam. Trends in gpcr drug discovery: new agents, targets and indications. *Nature reviews Drug discovery*, 16(12):829–842, 2017.

[7] Li-Huei Tsai, Ed Harlow, and Matthew Meyerson. Isolation of the human cdk2 gene that encodes the cyclin a-and adenovirus e1a-associated p33 kinase. *Nature*, 353(6340):174–177, 1991.

[8] Eric A Hanse, Christopher J Nelsen, Melissa M Goggin, Chelsea K Anttila, Lisa K Mullany, Cyril Berthet, Philipp Kaldis, Gretchen S Crary, Ryoko Kuriyama, and Jeffrey H Albrecht. Cdk2 plays a critical role in hepatocyte cell cycle progression and survival in the setting of cyclin d1 expression in vivo. *Cell Cycle*, 8(17):2802–2809, 2009.

[9] Peng Xia, Yuening Liu, Jingrui Chen, Shelby Coates, David X Liu, and Zhaokang Cheng. Inhibition of cyclin-dependent kinase 2 protects against doxorubicin-induced cardiomyocyte apoptosis and cardiomyopathy. *Journal of Biological Chemistry*, 293(51):19672–19685, 2018.

[10] Jonas Cicenas, Karthik Kalyan, Aleksandras Sorokinas, Edvinas Stankunas, Josh Levy, Ingrida Meskinyte, Vaidotas Stankevicius, Algirdas Kaupinis, and Mindaugas Valius. Roscovitine in cancer and other diseases. *Annals of translational medicine*, 3(10), 2015.

[11] Nicholas R Brown, Svitlana Korolchuk, Mathew P Martin, Will A Stanley, Rouslan Moukhametzianov, Martin EM Noble, and Jane A Endicott. Cdk1 structures reveal conserved and unique features of the essential cell cycle cdk. *Nature communications*, 6(1):1–12, 2015.

[12] Magd Badaoui, Pedro J Buigues, Dénes Berta, Gaurav M Mandana, Hankang Gu, Tamas Foldes, Callum J Dickson, Viktor Hornak, Mitsunori Kato, Carla Molteni, et al. Combined free-energy calculation and machine learning methods for understanding ligand unbinding kinetics. *Journal of chemical theory and computation*, 18(4):2543–2555, 2022.

[13] Doris A Schuetz, Wilhelmus Egbertus Arnout de Witte, Yin Cheong Wong, Bernhard Knasmueller, Lars Richter, Daria B Kokh, S Kashif Sadiq, Reggie Bosma, Indira Nederpelt, Laura H Heitman, et al. Kinetics for drug discovery: an industry-driven effort to target drug residence time. *Drug Discovery Today*, 22(6):896–911, 2017.

[14] M Bernetti, A Cavalli, and L Mollica. Protein–ligand (un) binding kinetics as a new paradigm for drug discovery at the crossroad between experiments and modelling. *MedChemComm*, 8(3):534–550, 2017.

[15] Yuzhen Niu, Shuyan Li, Dabo Pan, Huanxiang Liu, and Xiaojun Yao. Computational study on the unbinding pathways of b-raf inhibitors and its implication for the difference of residence time: insight from random acceleration and steered molecular dynamics simulations. *Physical Chemistry Chemical Physics*, 18(7):5622–5629, 2016.

[16] Alex Dickson and Samuel D Lotz. Multiple ligand unbinding pathways and ligand-induced destabilization revealed by wexplore. *Biophysical journal*, 112(4):620–629, 2017.

[17] Sun-Ting Tsai, En-Jui Kuo, and Pratyush Tiwary. Learning molecular dynamics with simple language model built upon long short-term memory neural network. *Nature communications*, 11(1):1–11, 2020.

[18] Frank Noé and Cecilia Clementi. Collective variables for the study of long-time kinetics from molecular trajectories: theory and methods. *Current opinion in structural biology*, 43:141–147, 2017.

[19] David J Huggins, Philip C Biggin, Marc A Dämgen, Jonathan W Essex, Sarah A Harris, Richard H Henchman, Syma Khalid, Antonija Kuzmanic, Charles A Laughton, Julien Michel, et al. Biomolecular simulations: From dynamics and mechanisms to computational assays of biological activity. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 9(3):e1393, 2019.

[20] Amin Ullah, Jamil Ahmad, Khan Muhammad, Muhammad Sajjad, and Sung Wook Baik. Action recognition in video sequences using deep bi-directional lstm with cnn features. *IEEE access*, 6:1155–1166, 2017.

[21] Matt W Gardner and SR Dorling. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15):2627–2636, 1998.

[22] Robert A Copeland, David L Pompliano, and Thomas D Meek. Drug–target residence time and its implications for lead optimization. *Nature reviews Drug discovery*, 5(9):730–739, 2006.

[23] Aravind Basavapathruni, Lei Jin, Scott R Daigle, Christina RA Majer, Carly A Therkelsen, Tim J Wigle, Kevin W Kuntz, Richard Chesworth, Roy M Pollock, Margaret P Scott, et al. Conformational adaptation drives potent, selective and durable inhibition of the human protein methyltransferase dot1l. *Chemical biology & drug design*, 80(6):971–980, 2012.

[24] Glenn S Van Aller, Melissa Baker Pappalardi, Heidi M Ott, Elsie Diaz, Martin Brandt, Benjamin J Schwartz, William H Miller, Dashyant Dhanak, Michael T McCabe, Sharad K Verma, et al. Long residence time inhibition of ezh2 in activated polycomb repressive complex 2. *ACS chemical biology*, 9(3):622–629, 2014.

[25] Kevin P Cusack, Ying Wang, Michael Z Hoemann, Jasmina Marjanovic, Roland G Heym, and Anil Vasudevan. Design strategies to address kinetics of drug binding and residence time. *Bioorganic & medicinal chemistry letters*, 25(10):2019–2027, 2015.

[26] Grant K Walkup, Zhiping You, Philip L Ross, Eleanor KH Allen, Fereidoon Daryaee, Michael R Hale, John O'donnell, David E Ehmann, Virna JA Schuck, Ed T Buurman, et al. Translating slow-binding inhibition kinetics into cellular and in vivo effects. *Nature chemical biology*, 11(6):416–423, 2015.

[27] Laidler, Keith J. transition-state theory. https://www.britannica.com/science/transition-state-theory, 2019. [Online; accessed 4-September-2022].

[28] Alan D McNaught, Andrew Wilkinson, et al. *Compendium of chemical terminology*, volume 1669. Blackwell Science Oxford, 1997.

[29] Ago Rinken, Santa Veiksina, and Sergei Kopanchuk. Dynamics of ligand binding to gpcr: Residence time of melanocortins and its modulation. *Pharmacological Research*, 113:747–753, 2016.

[30] Sidney Udenfriend, Louise Gerber, and Nathan Nelson. Scintillation proximity assay: a sensitive and continuous isotopic method for monitoring ligand/receptor and antigen/antibody interactions. *Analytical biochemistry*, 161(2):494–500, 1987.

[31] Simon G Patching. Surface plasmon resonance spectroscopy for characterisation of membrane protein–ligand interactions and its potential for drug discovery. *Biochimica et Biophysica Acta (BBA)-Biomembranes*, 1838(1):43–55, 2014.

[32] Panagiotis L Kastritis and Alexandre MJJ Bonvin. On the binding affinity of macromolecular interactions: daring to ask why proteins interact. *Journal of The Royal Society Interface*, 10(79):20120835, 2013.

[33] Rodrigo Casasnovas, Vittorio Limongelli, Pratyush Tiwary, Paolo Carloni, and Michele Parrinello. Unbinding kinetics of a p38 map kinase type ii inhibitor from metadynamics simulations. *Journal of the American Chemical Society*, 139(13):4780–4788, 2017.

[34] Susanta Haldar, Federico Comitani, Giorgio Saladino, Christopher Woods, Marc W van der Kamp, Adrian J Mulholland, and Francesco Luigi Gervasio. A multiscale simulation approach to modeling drug–protein binding kinetics. *Journal of chemical theory and computation*, 14(11):6093–6101, 2018.

[35] Massimiliano Bonomi, Davide Branduardi, Giovanni Bussi, Carlo Camilloni, Davide Provasi, Paolo Raiteri, Davide Donadio, Fabrizio Marinelli, Fabio Pietrucci, Ricardo A Broglia, et al. Plumed: A portable plugin for free-energy calculations with molecular dynamics. *Computer Physics Communications*, 180(10):1961–1972, 2009.

[36] Ludovico Sutto, Simone Marsili, and Francesco Luigi Gervasio. New advances in metadynamics. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 2(5):771–779, 2012.

[37] Zofia Trstanova. *Mathematical and algorithmic analysis of modified Langevin dynamics*. PhD thesis, Université Grenoble Alpes, 2016.

[38] Mikio Namiki. *Stochastic quantization*, volume 9. Springer Science & Business Media, 2008.

[39] Deqiang Zhang, Justin Gullingsrud, and J Andrew McCammon. Potentials of mean force for acetylcholine unbinding from the alpha7 nicotinic acetylcholine receptor ligand-binding domain. *Journal of the American Chemical Society*, 128(9):3019–3026, 2006.

[40] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2020.

[41] Tom M Mitchell and Tom M Mitchell. *Machine learning*, volume 1. McGraw-hill New York, 1997.

[42] Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[43] Anders Brahme. *Comprehensive biomedical physics*. Newnes, 2014.

[44] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[45] Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, 91(1), 1991.

[46] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[47] Peter J Angeline, Gregory M Saunders, and Jordan B Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE transactions on Neural Networks*, 5(1):54–65, 1994.

[48] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[49] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[50] Yoshua Bengio and Paolo Frasconi. Credit assignment through time: Alternatives to backpropagation. *Advances in neural information processing systems*, 6, 1993.

[51] C Lee Giles, Guo-Zheng Sun, Hsing-Hen Chen, Yee-Chun Lee, and Dong Chen. Higher order recurrent networks and grammatical inference. *Advances in neural information processing systems*, 2, 1989.

[52] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[53] Wikipedia contributors. Recurrent neural network — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Recurrent_neural_network&oldid=1095187237, 2022. [Online; accessed 4-August-2022].

[54] Jürgen Schmidhuber, Daan Wierstra, and Faustino J Gomez. Evolino: Hybrid neuroevolution/optimal linear search for sequence prediction. In *Proceedings of the 19th International Joint Conferenceon Artificial Intelligence (IJCAI)*, 2005.

[55] Santiago Fernández, Alex Graves, and Jürgen Schmidhuber. An application of recurrent neural networks to discriminative keyword spotting. In *International Conference on Artificial Neural Networks*, pages 220–229. Springer, 2007.

[56] Anvita Gupta, Alex T Müller, Berend JH Huisman, Jens A Fuchs, Petra Schneider, and Gisbert Schneider. Generative recurrent networks for de novo drug design. *Molecular informatics*, 37(1-2):1700111, 2018.

[57] Tianjun Zhang, Shuang Song, Shugang Li, Li Ma, Shaobo Pan, and Liyun Han. Research on gas concentration prediction models based on lstm multidimensional time series. *Energies*, 12(1):161, 2019.

[58] Rich Wolski, John Brevik, Ryan Chard, and Kyle Chard. Probabilistic guarantees of execution duration for amazon spot instances. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–11, 2017.

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[60] Igor Chikalov, Peggy Yao, Mikhail Moshkov, and Jean-Claude Latombe. Learning probabilistic models of hydrogen bond stability from molecular dynamics simulation trajectories. *BMC bioinformatics*, 12(1):1–6, 2011.

[61] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[62] Bin Jiang, Jun Li, and Hua Guo. Potential energy surfaces from high fidelity fitting of ab initio points: the permutation invariant polynomial-neural network approach. *International Reviews in Physical Chemistry*, 35(3):479–506, 2016.

[63] Jie Cui and Roman V Krems. Gaussian process model for collision dynamics of complex molecules. *Physical review letters*, 115(7):073202, 2015.

[64] Florian Häse, Stéphanie Valleau, Edward Pyzer-Knapp, and Alán Aspuru-Guzik. Machine learning exciton dynamics. *Chemical Science*, 7(8):5139–5147, 2016.

[65] Oliver Fleetwood, Marina A Kasimova, Annie M Westerlund, and Lucie Delemotte. Molecular insights from conformational ensembles via machine learning. *Biophysical Journal*, 118(3):765–780, 2020.

[66] Chitrak Gupta, John Kevin Cava, Daipayan Sarkar, Eric Wilson, John Vant, Steven Murray, Abhishek Singharoy, and Shubhra Kanti Karmaker. Mind reading of the proteins: Deep-learning to forecast molecular dynamics. *bioRxiv*, 2020.

[67] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[68] Manu Joseph. Interpretability part 3: opening the black box with lime and shap, 2020.