**REINDEER: A Protein-Ligand Feature Generator Software for Machine Learning Algorithms**

Milad Rayka1\*

1. Unaffiliated

Corresponding author:

Milad Rayka, milad.rayka@yahoo.com

**Abstract**

**Keywords:** Cheminformatics, Protein-Ligand Complex, Software, Machine Learning, Feature Engineering

1- Introduction

In the past decade, we have witnessed a proliferation of data-driven approaches for designing protein-ligand scoring functions, which scientists apply to predict the binding affinity of a protein-ligand complex. These new scoring functions employ traditional machine learning (ML) and deep learning algorithms (DL) for deriving a non-linear relationship between binding affinity quantity and a representation of the protein-ligand complex [1-9].

A designed scoring function can be evaluated based on its ability to perform four tasks: scoring, ranking, docking, and screening. The first two tasks involve predicting binding affinity values, while the other two tasks evaluate the scoring function's ability to distinguish between native or near-native poses and decoys, as well as true binders from non-binders [10, 11]. The PDBbind dataset [12], DUD-E [13], and DEKOIS2.0 [14] are among the popular datasets for training and testing, in spite of the fact that there are some hidden biases in these datasets [15, 16].

In the ML case, it is required to represent a protein-ligand complex in terms of a feature vector by applying feature engineering techniques, while in the DL case, these representations are mostly learned end-to-end during the training phase.

RF-Score [17], ECIF∷LD-GBT [18], and multi-shelled ECIF [19] are ML-based scoring functions that exploit the occurrence of interatomic contacts - also ligand descriptors in ECIF∷LD-GBT case - to generate a feature vector for ML algorithms like Random Forest (RF) or Gradient Boosting Decision Trees (GBT). ET-Score [20] and its’ enhanced versions, i.e., GB-Score [21] and ENS-Score [22], employed distance-weighted interatomic contacts and Extremely Randomized Trees (ERT) for constructing an ML-based scoring function. ∆vinaRF20 [23], ∆vinaXGB [24], and ∆LinF9XGB [25] employ some Autodock Vina empirical terms alongside other descriptors, e.g., RDKit descriptors for ligand, for feature generation procedures, and RF and eXtreme Gradient Boosting (XGB) algorithms as learners. Proteo-chemometrics IFPs [26] and PLEC-FP [27] apply different generalized versions of extended-connectivity fingerprints (ECFPs) for protein-ligand representation. SMPLIP-Score [28] utilizes a featurization method to vectorize and embed the interaction fingerprint pattern between the ligand‑binding site environment and fragments of ligands. TB-IECS [29] employs some theory-based energy terms, e.g., van der Waals, to generate a representation for an XGB algorithm. OPRC-GBT [30], AGL-Score [31], PPS-ML [32], EISA-Score [33], hypergraph-based persistent cohomology (HPC) model [34], and PerSpect ML [35] are examples of ML-based scoring functions that utilize topology principles and invariants for generating feature vectors for a protein-ligand complex.

Unlike ML algorithms, DL methods have the capability to learn an end-to-end representation of a protein-ligand complex. Seo et al. [36], AEScore [37], and BCL-AffinityNet [38] are among the recent examples of DL-based scoring functions that utilize different variants of Feed-Forward Neural Networks (FFNNs). For the recent examples of Convolutional Neural Networks (CNNs) –based scoring functions, we can mention Jones et al. [39], DeepDTAF [40], AA-Score [41], and OnionNet-2 [42]. GIGN [43], PIGNet2 [44], CarsiDock [45], and GAABind [46] can be categorized as instances of Graph Neural Networks (GNNs) –based binding affinity estimators.

Although we have witnessed a progression in designing ML and DL-based scoring functions in the past few years, diverse feature engineering methods of these scoring functions are buried in papers and their accompanying source codes and are not available for other researchers or users, and making it difficult for others to access and use them.

Descriptor Data Bank (DDB) [47] was one of the first attempts to make feature engineering techniques accessible for the end user. It contained 16 descriptor extraction methods for proteins, ligands, and protein-ligand complexes, alongside several machine learning toolboxes, e.g., for filtering irrelevant features. DDB was deployed as a website that is currently unavailable, and the accompanying source code for the published paper is missing. Open Drug Discovery Toolkit (ODDT) is a Python package [48] that provides tools for drug discovery, including molecular docking and virtual screening, to a broader audience. ODDT provides RF-Score, PLEC, and NNScore [49] as ML-based scoring functions. Despite the usefulness of ODDT, it seems that its’ development is stalled. ML-based protein-ligand interaction capturer (ML-PLIC) is a web platform [50], an enhanced version of Artificial Intelligence based Scoring Function Platform (ASFP) [51], to automatically design new ML-based scoring function. ML-PLIC provides feature vectors from PLEC, SPLIF [52], and NNScore scoring functions.

In this manuscript, we introduce p**r**ot**e**in-l**i**ga**nd** f**e**ature g**e**ne**r**ator (REINDEER) software to make protein-ligand representation techniques accessible to end-users (Figure 1). Due to this, REINDEER includes eight geometry and topology-based feature engineering methods that have been published previously. These methods are selected from RF-Score [17], ET-Score [20], ECIF∷LD-GBT [18], OnionNet [53], PPS-ML [35], EISA-Score [33], OPRC-GBT [30], and HPC [34] ML or DL-based scoring functions. In the first release of this software, we only focus on methods that incorporate geometry or topology concepts for feature generation and exclude other famous methods, e.g., ∆vinaRF20, that use energy-based or empirical terms for protein-ligand representations. Also, we chose these methods because they have high-quality GitHub repositories and well-written papers, which makes implementing these methods feasible. REINDEER provides a Command Line Interface (CLI) and a Graphical User Interface (GUI) to make it more user-friendly. REINDEER is developed in Python programming language, with the mindset of minimum dependency on other packages and providing parallelization for faster feature generation. To evaluate the utility of REINDEER, we designed several ML-based scoring functions by using provided feature generation methods in REINDEER. To this end, we employed XGB as a learner and PDBbind v2020 dataset. REINDEER software is available a thttps://github.com/miladrayka/reindeer\_software.

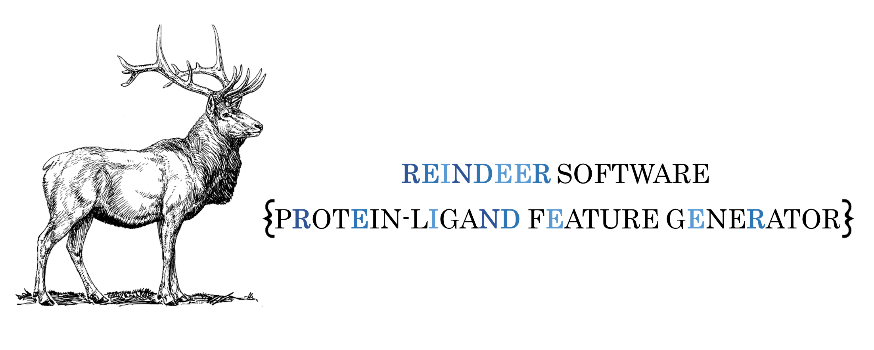


Figure 1- Logo of REINDEER software.

The paper is organized as follows. In section 2, a summary of different protein-ligand feature generation methods is provided. Details of the implementation of REINDEER are gathered in section 3. Section 4 is devoted to a case study to evaluate the utility of REINDEER software. We conclude our paper in section 5.

2- Theory

This section provides a brief explanation of each feature generation method. As mentioned before, in the current version of REINDEER, only geometry and topology-based methods are selected. A summary of these methods is depicted in Table 1.

Table 1- A summary of implemented feature generation methods in REINDEER.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Scoring Function | Feature generation method | Dimensions of feature vector | Reference |
| 1 | RF-Score | Occurrence of Interatomic Contact | 50 | [17] |
| 2 | ET-Score | Distance-Weighted Interatomic Contact | 200 | [20] |
| 3 | ECIF∷LD-GBT | Extended Connectivity Interaction Features | 1540 | [18] |
| 4 | OnionNet-2 | Multi-Shell Occurrence of Interatomic Contact | 10416 | [42] |

Occurrence of Interatomic Contact

Occurrence of interatomic contact (OIC) was introduced by Ballester et al. for developing the RF-Score scoring function [17]. In this technique, the authors represented a protein-ligand complex by counting the number of occurrences of a specific pair of protein and ligand atoms below a distance threshold. Ten elemental atom types (H, C, O, N, F, P, S, Cl, Br, and I) were allocated for protein and ligand, although in its original implementation, the Hydrogen element was omitted. The following formula calculates the occurrence (1):

|  |  |
| --- | --- |
|  | (1) |

Where xi,j is the number of contacts between i and j atom types. k and l are protein and ligand atoms belonging to the i and j atom types. dkl is the Euclidean distance between k and l atoms, and Θ is the Heaviside step function that counts contacts below dcutoff=12 Å. The feature vector contains 50 integer-valued features.

Distance-Weighted Interatomic Contact

Distance-weighted interatomic contact (DWIC) was employed by ET-Score [20] for vectorial representation of a protein-ligand complex and further used by GB-Score [21] and ENS-Score [22]. Like OIC, ten elemental atom types are chosen for both ligand and protein. However, protein atom types are further augmented by considering the nature of amino acid side chains. To reflect the different characteristics of amino acids, they are classified into four groups (Charged (c), Polar (p), Amphipathic (a), and Hydrophobic (h)):

Charged={Arg, Lys, Asp, Glu}

Polar={Gln, Asn, His, Ser, Thr, Cys}

Amphipathic={Trp, Tyr, Met}

Hydrophobic={Ile, Leu, Phe, Val, Pro, Gly, Ala}

Therefore, each elemental protein atom type belongs to four groups. As an example, CP denotes a carbon atom of polar residues. At its’ core, DWIC is similar to OIC in the feature generation by considering atom types pair of protein and ligand, but, in DWIC, the Heaviside step function is replaced by a function, which differentiated close and distant interatomic contacts by applying an inverse-square factor. The following equation (2) describes this function:

|  |  |
| --- | --- |
|  | (2) |

The definition of symbols is similar to equation (1), and like OIC, dcutoff=12Å is applied. The DIWC feature vector includes 200 float-valued features.

Extended Connectivity Interaction Features

Extended Connectivity Interaction Features, or concisely ECIF, share the same essence, like DIWC, with OIC [18]. However, ECIF applies different atom type representations and dcutoff value (6Å) and only retains the counting atom type pairs scheme of OIC. The authors defined atom types in ECIF by considering the atom environment. Atom symbol, explicit valence, number of attached heavy atoms, number of attached hydrogens, aromaticity, and ring membership were employed for this definition, which results in 22 and 70 atom types for protein and ligand, respectively. The final feature vector comprises 1540 integer-valued features.

Multi-Shell Occurrence of Interatomic Contact

Multi-Shell Occurrence of Interatomic Contact (MS-OIC) is a technique used by OnionNet-2 [42] and OnionNet [53] for generating features. Similar to other mentioned feature generation methods, MS-OIC counts the occurrence of various entity pairs in proteins and ligands. Eight atom types are selected for ligands: H, C, N, O, P, S, HAL, and DU. HAL represents all halogen atoms, and DU represents all other atoms except the mentioned ones. On the protein side, twenty natural amino acids plus “OTH” symbols are considered as entities. This new symbol is used to represent water, ions, and non-standard amino acids. To represent protein-ligand in more detail, residue-atom pair contacts are counted in 62 constructed shells with different radii around the ligand, of which each shell thickness is 0.5 Å. The final feature vector has a dimension of 10416 (8 × 21 × 62).

3- Method

Implementation

Python and its’ library

Quality by black pylint

Pip and conda

Cli default parameters from papers, gui, streamlit, tutorial for cli, gui

Google colab notebook inspired from colabfold

4- Case Study

We provide the following case study to demonstrate the functionalities of REINDEER. In this case study, several ML-based scoring functions are trained by utilizing XGB as a learner, PDBbind refined set v2020 as a dataset, and eight distinct feature generation methods of REINDEER software.

PDBbind dataset is one of the commonly used datasets in designing scoring functions for protein-ligand complexes [12]. PDBbind dataset for protein-ligand complexes includes general, refined, and core sets. Structures and binding affinity values of the core set have the highest quality and are used as a benchmark test set in a comparative assessment of scoring function (CASF) benchmark. The structures in general and refined sets are usually utilized as training sets, although this is not suggested by the curator of the PDBbind dataset [54]. In this case study, we used PDBbind refined set v2020, in which CASF core set v2016 structures are excluded from it, and CASF core set v2016 as the test set. Further, we sampled 300 data points from the train set as a validation set. Details of the refined set v2020 are discussed in our previous paper. Train, validation, and test sets have 4750, 300, and 285 members, respectively. Table S1 reports the PDBIDs of these sets.

At the preprocessing step, we discard static, quasi-static, and correlated features, which we define as features with zero variance, less than 0.01 variance, and above 0.95 correlation, respectively. We employ the XGB algorithm of the XGBoost Python package [55] as a learner to find a relationship between generated features and binding affinity values. We use Optuna [56] to optimize the hyperparameters of the XGB regression algorithm on the validation set. Table S2 depicts the interested hyperparameters along their search space. The performance of scoring functions is reported based on root-mean-square error (RMSE) and Pearson's correlation coefficient (RP) metrics. The reported metrics are based on the average of five distinct trained models, each trained using different random numbers.

comparison

5- Conclusion

6- References

1. Ain QU, Aleksandrova A, Roessler FD, Ballester PJ. Wiley Interdiscip Rev: Comput Mol Sci 5, 6 (2015): 405-424.

2. Colwell LJ. Curr Opin Struct Biol 49, (2018): 123-128.

3. Shen C, Ding J, Wang Z, Cao D, Ding X, Hou T. Wiley Interdiscip Rev: Comput Mol Sci 10, 1 (2020): e1429.

4. Li H, Sze KH, Lu G, Ballester PJ. Wiley Interdiscip Rev: Comput Mol Sci 11, 1 (2021): e1478.

5. Pellicani F, Dal Ben D, Perali A, Pilati S. Molecules 28, 4 (2023): 1661.

6. Yang C, Chen EA, Zhang Y. Molecules 27, 14 (2022): 4568.

7. Schapin N, Majewski M, Varela-Rial A, Arroniz C, De Fabritiis G. Artificial Intelligence Chemistry 1, 2 (2023): 100020.

8. Hagg A, Kirschner KN. J Chem Inf Model 63, 15 (2023): 4505-4532.

9. Zhang Y, Li S, Meng K, Sun S. J Chem Inf Model 64, 5 (2024): 1456–1472.

10. Li Y, Liu Z, Li J, Han L, Liu J, Zhao Z, Wang R. J Chem Inf Model 54, 6 (2014): 1700-1716.

11. Li Y, Han L, Liu Z, Wang R. J Chem Inf Model 54, 6 (2014): 1717-1736.

12. Su M, Yang Q, Du Y, Feng G, Liu Z, Li Y, Wang R. J Chem Inf Model 59, 2 (2019): 895-913.

13. Mysinger MM, Carchia M, Irwin JJ, Shoichet BK. J Med Chem 55, 14 (2012): 6582-6594.

14. Bauer MR, Ibrahim TM, Vogel SM, Boeckler FM. J Chem Inf Model 53, 6 (2013): 1447-1462.

15. Shen C, Hu Y, Wang Z, Zhang X, Pang J, Wang G, Zhong H, Xu L, Cao D, Hou T. Briefings Bioinf 22, 3 (2021): bbaa070.

16. Li J, Guan X, Zhang O, Sun K, Wang Y, Bagni D, Head-Gordon T. ArXiv (2023).

17. Ballester PJ, Mitchell JB. Bioinformatics 26, 9 (2010): 1169-1175.

18. Sánchez-Cruz N, Medina-Franco JL, Mestres J, Barril X. Bioinformatics 37, 10 (2021): 1376-1382.

19. Shiota K, Akutsu T. Bioinf Adv 3, 1 (2023): vbad155.

20. Rayka M, Karimi‐Jafari MH, Firouzi R. Mol Inf 40, 8 (2021): 2060084.

21. Rayka M, Firouzi R. Mol Inf 42, 3 (2023): 2200135.

22. Rayka M, Mirzaei M, Mohammad Latifi A. Mol Inf (2024): e202300292.

23. Wang C, Zhang Y. J Comput Chem 38, 3 (2017): 169-177.

24. Lu J, Hou X, Wang C, Zhang Y. J Chem Inf Model 59, 11 (2019): 4540-4549.

25. Yang C, Zhang Y. J Chem Inf Model 62, 11 (2022): 2696-2712.

26. Wang DD, Xie H, Yan H. Bioinformatics 37, 17 (2021): 2570-2579.

27. Wójcikowski M, Kukiełka M, Stepniewska-Dziubinska MM, Siedlecki P. Bioinformatics 35, 8 (2019): 1334-1341.

28. Kumar S, Kim M-h. J Cheminf 13, 28 (2021): 1-17.

29. Zhang X, Shen C, Jiang D, Zhang J, Ye Q, Xu L, Hou T, Pan P, Kang Y. J Cheminf 15, 1 (2023): 63.

30. Wee J, Xia K. J Chem Inf Model 61, 4 (2021): 1617-1626.

31. Nguyen DD, Wei G-W. J Chem Inf Model 59, 7 (2019): 3291-3304.

32. Liu R, Liu X, Wu J. J Chem Inf Model 63, 3 (2023): 1066-1075.

33. Rana MM, Nguyen DD. J Chem Inf Model 62, 18 (2022): 4329-4341.

34. Liu X, Wang X, Wu J, Xia K. Briefings Bioinf 22, 5 (2021): bbaa411.

35. Meng Z, Xia K. Sci Adv 7, 19 (2021): eabc5329.

36. Seo S, Choi J, Park S, Ahn J. BMC Bioinf 22, 542 (2021): 1-15.

37. Meli R, Anighoro A, Bodkin MJ, Morris GM, Biggin PC. J Cheminf 13, 1 (2021): 59.

38. Brown BP, Mendenhall J, Geanes AR, Meiler J. J Chem Inf Model 61, 2 (2021): 603-620.

39. Jones D, Kim H, Zhang X, Zemla A, Stevenson G, Bennett WD, Kirshner D, Wong SE, Lightstone FC, Allen JE. J Chem Inf Model 61, 4 (2021): 1583-1592.

40. Wang K, Zhou R, Li Y, Li M. Briefings Bioinf 22, 5 (2021): bbab072.

41. Pan X, Wang H, Zhang Y, Wang X, Li C, Ji C, Zhang JZ. J Chem Inf Model 62, 10 (2022): 2499-2509.

42. Wang Z, Zheng L, Liu Y, Qu Y, Li Y-Q, Zhao M, Mu Y, Li W. Front Chem 9, (2021): 753002.

43. Yang Z, Zhong W, Lv Q, Dong T, Yu-Chian Chen C. J Phys Chem Lett 14, 8 (2023): 2020-2033.

44. Moon S, Hwang S-Y, Lim J, Kim WY. Digital Discovery 3, 2 (2024): 287-299.

45. Cai H, Shen C, Jian T, Zhang X, Chen T, Han X, Yang Z, Dang W, Hsieh C-Y, Kang Y. Chem Sci 15, 4 (2024): 1449-1471.

46. Tan H, Wang Z, Hu G. Briefings Bioinf 25, 1 (2024): bbad462.

47. Ashtawy HM, Mahapatra NR. J Chem Inf Model 58, 1 (2018): 134-147.

48. Wójcikowski M, Zielenkiewicz P, Siedlecki P. J Cheminf 7, (2015): 1-6.

49. Durrant JD, McCammon JA. J Chem Inf Model 51, 11 (2011): 2897-2903.

50. Zhang X, Shen C, Wang T, Deng Y, Kang Y, Li D, Hou T, Pan P. Briefings Bioinf 24, 5 (2023): bbad295.

51. Zhang X, Shen C, Guo X, Wang Z, Weng G, Ye Q, Wang G, He Q, Yang B, Cao D. J Cheminf 13, 6 (2021): 1-9.

52. Da C, Kireev D. J Chem Inf Model 54, 9 (2014): 2555-2561.

53. Zheng L, Fan J, Mu Y. ACS omega 4, 14 (2019): 15956-15965.

54. Su M, Feng G, Liu Z, Li Y, Wang R. J Chem Inf Model 60, 3 (2020): 1122-1136.

55. Chen T, Guestrin C: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining: 2016. (Year): 785-794.

56. Akiba T, Sano S, Yanase T, Ohta T, Koyama M: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining: 2019. (Year): 2623-2631.