BMC Bioinformatics

SOFTWARE

Open Access

Reindeer: a protein-ligand feature generator software for machine learning algorithms

Milad Rayka[1*] and S. Shahab Naghavi[1]

*Correspondence:
Milad Rayka
miladrayka93@gmail.com
[1]Department of Physical and Computational Chemistry, Shahid Beheshti University, 1983969411, Evin, Tehran, Iran

Abstract

Background Feature generation drives machine learning-based drug discovery by producing unique, compact, invariant, and computationally efficient descriptors. In structural protein–ligand binding affinity prediction, machine learning has developed a range of feature generation methods, with geometry-based techniques achieving the highest predictive accuracy—particularly when used with the eXtreme Gradient Boosting (XGBoost) algorithm. Unfortunately, these techniques remain largely inaccessible to end-users due to their reliance on programming expertise. To address this, we introduce `REINDEER`, a software tool that streamlines geometry-based feature generation through an intuitive Graphical User Interface (GUI), Command Line Interface (CLI), and Python API.

Results `REINDEER` implements four state-of-the-art feature generation methods: Occurrence of Interatomic Contact (OIC), Distance-Weighted Interatomic Contact (DWIC), Extended Connectivity Interaction Feature (ECIF), and Multi-Shell Occurrence of Interatomic Contact (MS-OIC). The software also supports parallel processing to accelerate computation. To demonstrate the `REINDEER`'s capabilities, and validate its implementations, we evaluate it on a novel protein-peptide dataset, assessing its performance in three key areas: (i) computational speed under CPU parallelization, (ii) binding affinity prediction accuracy using XGBoost, and (iii) uncertainty quantification via conformal prediction. Results confirm the computational efficiency of the implemented methods and demonstrate that the generated features produce predictive models in a practical workflow. Among the four methods, ECIF paired with XGBoost yields the highest Pearson's correlation coefficient ($\mathrm{R}_P = 0.651$). Furthermore, the XGBoost-ECIF conformal predictor proves both valid (mean accuracy of 81%) and efficient (mean confidence region size of 1.442).

Conclusions By democratizing access to advanced feature generation methods through GUI, CLI, and Python API, `REINDEER` enables researchers with limited programming expertise to harness cutting-edge machine learning techniques in designing machine learning-based scoring functions for their specific problems.

Keywords Machine learning, Feature engineering, Protein-ligand binding affinity, Software, Scoring function, Conformal prediction, Uncertainty quantification

Background

Traditional machine learning (hereafter ML) and deep learning (DL) have revolutionized protein-ligand binding affinity predictions, a critical task usually accomplished by scoring functions [1–4]. While deep learning models—such as Deep Neural Networks [5], Graph Neural Networks[6], and Transformers [7]—have gained recent popularity, they present challenges including limited interpretability[8] (i.e., "black box" nature) and a strong dependence on large training datasets [9]. Deep learning methods also require substantial computational resources, particularly GPU acceleration for both training and inference. In contrast, ML-based scoring functions offer competitive performance with reduced data requirements and greater model transparency. However, this comes at the cost of explicit feature engineering: ML models require carefully crafted descriptors [10, 11], and generating high-dimensional geometry-based features can itself be computationally intensive. However, unlike deep learning, this feature generation cost is one-time, and the resulting static feature vectors can be reused across multiple modeling tasks. For instance, ECIF::LD-GBT [12] and ENS-Score[13] both use interatomic feature engineering along with Gradient Boosting Decision Trees (GBDT) [14] to get results on par with computationally intensive deep learning methods on CASF-2016 benchmark. Given the scarcity of protein-ligand complexes with binding affinity data, ML stands out as a practical choice, assuming that feature engineering becomes more streamlined.

Despite their undeniable practicality, feature engineering methods are often inaccessible, buried within academic papers and their accompanying code. The Descriptor Data Bank (DDB) was one of the earliest efforts to make these techniques accessible to end users [15]. The DDP provides 16 descriptor extraction methods for proteins, ligands, and protein-ligand complexes, along with various ML toolboxes to filter out irrelevant features. Unfortunately, the DDB website and its source code are no longer available. Likewise, the Open Drug Discovery Toolkit (ODDT) [16], a Python package for drug discovery, showed early promise with tools for molecular docking, virtual screening, and machine-learning-based scoring functions (RF-Score [17], PLEC [18], and NNScore [19]). However, its development has stopped. More recently, the ML-based Protein-Ligand Interaction Capturer (ML-PLIC) [20], an upgraded iteration of the Artificial Intelligence-based Scoring Function Platform (ASFP) [21], was designed to automatically generate new ML-based scoring functions. This platform—offering feature vectors from established scoring functions like PLEC, SPLIF [22], and NNScore— streamlines the feature generation process. However, it is solely accessible through a web application, and it remains far from the ideal solution. A structured comparison between the mentioned tools and the current work is gathered in Table S1.

While modern protein and molecular language models (e.g., ESM2 [23], and ChemBERTa [24]) provide powerful end-to-end representations, the geometry-based descriptors have distinct practical and scientific advantages. First, they are inherently interpretable: each feature corresponds directly to a physical interaction, enabling straightforward attribution of predictions to structural components. Second, they are computationally lightweight and do not require GPU acceleration or large pretrained models, making them accessible in resource-constrained environments. Third, they perform robustly in low-data regimes, where large fine-tuning datasets may be unavailable. Finally, these fixed-length descriptors integrate seamlessly into established machine-learning workflows, supporting rapid prototyping and feature selection [25, 26].

To address this gap, we present REINDEER (**p**rot**e**in-li**g**a**nd** f**e**ature g**e**ne**r**ator): an open-source, multi-interface software platform that streamlines the generation of geometry-based protein-ligand features. REINDEER integrates four well-established geometry-based feature engineering methods from RF-Score, ET-Score [27, 28], ECIF::LD-GBT, and OnionNet-2 [29]. The current release (version 0.2.0) focuses solely on geometry-based methods, excluding energy-based or empirical approaches like $\Delta_{Vina}RF_{20}$[30]. It's worth mentioning REINDEER enhances user-friendliness through both Command Line and Graphical User Interfaces (CLI and GUI). Developed in Python with minimal dependencies, it also supports parallelization for accelerated computation.

To demonstrate its capabilities, we use REINDEER to design an ML-based scoring function for a recently published peptide-protein dataset[31] (see Fig. 1). Peptides, as a new modality, open promising therapeutic avenues, especially as conventional small-molecule drugs reach their discovery limits [32, 33]. To this end, we opt for the XGBoost algorithm [34] alongside the Optuna package [35] for learning and hyperparameter optimization, respectively. We also augment our ML-based scoring function by incorporating Conformal Prediction (CP) [36–38], which quantifies uncertainty and provides confidence estimates for predicted binding affinities. Our results demonstrate that REINDEER accelerates drug design by generating reliable and well-optimized scoring functions for peptide-protein complexes.

Implementation

Feature engineering techniques

In this version of REINDEER, we have implemented only geometry-based methods. As listed in Table 1 and Fig. S1, we select the Occurrence of Interatomic Contact (OIC) from RF-Score [17], the Distance-Weighted Interatomic Contact (DWIC) from ET-Score [27], the Extended Connectivity Interaction Features (ECIF) from ECIF::LD-GBDT [12],

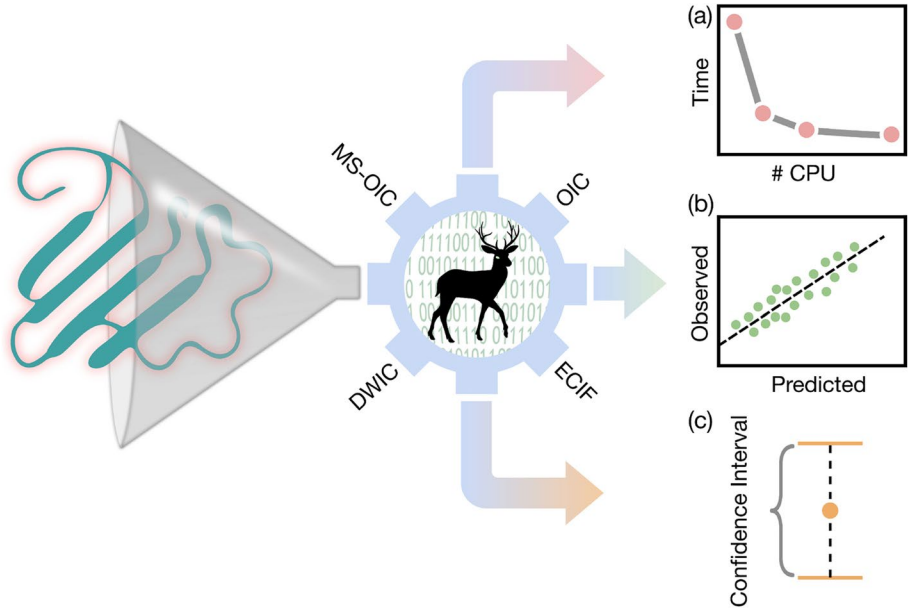

Fig. 1 Overview of the workflow in this study. The workflow depicts the generation of OIC, DWIC, ECIF, and MS-OIC by REINDEER software. The software has been assessed at three levels: **a** execution time in parallel computation, **b** designing an ML-based scoring function for the protein-peptide dataset, and **c** conformal prediction for uncertainty quantification

Table 1 A summary of implemented feature generation methods in REINDEER.1_A is the indicator function. D is a set of all pairwise distances. i and N indicate the shell index and the number of shells in MS-OIC

SF	Feature	Dim	Formula	Difference to OIC
RF-Score	Occurrence of Interatomic Contact	50	$1_A : D \to \{0,1\} A \ = \ \{d_{lk} \vert d_{lk} \leqslant d_{\mathrm{cutoff}}, d_{lk} \in D\}$	–
ET-Score	Distance-Weighted Interatomic Contact	200	$f_A := \frac{1}{d_{lk}^2} f_A : D \to \mathbb{R} A \ = \ \{d_{lk} \vert d_{lk} \leqslant d_{\mathrm{cutoff}}, d_{lk} \in D\}$	Extended atom types by incorporating amino acids and weighted features
ECIF::LD-GBDT	Extended Connectivity Interaction Features	1540	$1_A : D \to \{0,1\} A \ = \ \{d_{lk} \vert d_{lk} \leqslant d_{\mathrm{cutoff}}, d_{lk} \in D\}$	Extended atom types based on atomic enviroments
OnionNet-2	Multi-Shell Occurrence of Interatomic Contact	10416	$1_A : D \to \{0,1\} A \ = \ \{d_{lk} \vert (i-2)\delta + d_0 \leq d_{lk} < (i-1)\delta + d_0, d_{lk} \in D \text{ and } i \in N\}$	Extended atom types through concenteric shells

and the Multi-Shell Occurrence of Interatomic Contact (MS-OIC) from OnionNet-2 [29]. We pick these feature vectors for their simplicity, popularity, and strong performance compared to leading deep learning-based scoring functions, each of which we explore in detail in the following section.

Occurrence of interatomic contact (OIC)

OIC introduced by Ballester et al. [17], forms the foundation of the RF-Score. This method represents a protein-ligand complex by counting how often specific atom pairs occur within a set distance threshold. The method assigns ten elemental atom types (H, C, O, N, F, P, S, Cl, Br, and I) to protein and ligand, though hydrogen was originally excluded. The occurrence is computed as:

$$x_{i,j} = \sum_{k=1}^{K_j} \sum_{l=1}^{L_i} \Theta \left(d_{\text{cutoff}} - d_{kl} \right) \tag{1}$$

where $x_{i,j}$ is the number of contacts between *i* and *j* atom types. *k* and *l* belong to protein and ligand, classified as *i* and *j*, respectively. d_{kl} is their Euclidean distance, and Θ is the Heaviside step function, which counts contacts within the $\mathrm{d}_{cutoff} = 12$ Å.

Distance-weighted interatomic contact (DWIC)

DWIC, used in ET-Score [27] to represent protein-ligand complexes as vectors, also forms the foundation of GB-Score [28] and ENS-Score [13]. Like OIC, it defines ten elemental atom types for both ligand and protein. However, it refines protein atom types based on the properties of amino acid side chains, grouping them into four categories to capture their distinct characteristics: Charged (c), Polar (p), Amphipathic (a), and Hydrophobic (h).

$$\begin{aligned} \text{Charged} &= \{\text{Arg, Lys, Asp, Glu}\} \\ \text{Polar} &= \{\text{Gln, Asn, His, Ser, Thr, Cys}\} \\ \text{Amphipathic} &= \{\text{Trp, Tyr, Met}\} \\ \text{Hydrophobic} &= \{\text{Ile, Leu, Phe, Val, Pro, Gly, Ala}\} \end{aligned}$$

Thus, each elemental type of protein atom fits into one of four groups. For instance, C_p represents a carbon atom in a polar residue. Like OIC, DWIC builds features from protein-ligand atom-type pairs. But, instead of using the Heaviside step function, DWIC weighs interatomic contacts by the inverse square of their distance. The following equation defines this rule:

$$x_{i,j} = \sum_{k=1}^{K_j} \sum_{l=1}^{L_i} \frac{1}{d_{lk}^2} \tag{2}$$

The symbols follow the same definitions as in Eq. 1. As with OIC, DWIC sets $\mathrm{d}_{cutoff} = 12$ Å.

Extended connectivity interaction features (ECIF)

ECIF shares the same fundamental principles as DWIC and OIC but differs in atom type representation and cutoff distance ($\mathrm{d}_{cutoff} = 6$ Å) [12]. Retaining only OIC's counting

Table 2 Overview of four protein–ligand featurization algorithms. Each algorithm processes atomic features and spatial distances to generate interaction-based descriptors

OIC	DWIC
Input Protein, Ligand, $d_{\text{cutoff}} = 12$ Å	*Input* Protein, Ligand, ResidueGroups, $d_{\text{cutoff}} = 12$ Å
Output Feature vector *x*[*i*][*j*]	*Output* Feature vector *x*[*i*][*j*]
1: $x[i][j] \leftarrow 0$	1: $x[i][j] \leftarrow 0$
2: For all protein atom *p*: $i \leftarrow$ type of *p*	2: For all protein atom *p*: $i \leftarrow$ (element, group of *p*)
3: For all ligand atom *l*: $j \leftarrow$ type of *l*, $d \leftarrow \text{dist}(p, l)$	3: For all ligand atom *l*: $j \leftarrow$ type of *l*, $d \leftarrow \text{dist}(p, l)$
4: If $d \leq d_{\text{cutoff}}$: $x[i][j] \leftarrow x[i][j] + 1$	4: If $d \leq d_{\text{cutoff}}$: $x[i][j] \leftarrow x[i][j] + \frac{1}{d^2}$
5: Return *x*	5: Return *x*
ECIF	*MS-OIC*
Input Protein, Ligand, $d_{\text{cutoff}} = 6$ Å	*Input* Protein, Ligand, $d_{\text{cutoff}} = 31$ Å, ShellWidth = 0.5 Å
Output Feature vector *x*[*i*][*j*]	*Output* Feature tensor *x*[*i*][*j*][*s*]
1: $x[i][j] \leftarrow 0$	1: $x[i][j][s] \leftarrow 0$
2: For all protein atom *p*: $i \leftarrow \text{ECIFType}(p)$	2: For all ligand atom *l*: $i \leftarrow$ ligand type
3: For all ligand atom *l*: $j \leftarrow \text{ECIFType}(l)$, $d \leftarrow \text{dist}(p, l)$	3: For all protein atom *p*: $j \leftarrow$ residue type, $d \leftarrow \text{dist}(p, l)$
4: If $d \leq d_{\text{cutoff}}$: $x[i][j] \leftarrow x[i][j] + 1$	4: If $d \leq d_{\text{cutoff}}$: $s \leftarrow \lfloor d/\text{ShellWidth} \rfloor + 1$
5: Return *x*	5: If $1 \leq s \leq 62$: $x[i][j][s] \leftarrow x[i][j][s] + 1$
	6: Return *x*

scheme, ECIF classifies atoms by their environments—using the atom symbol, explicit valence, number of attached heavy atoms, number of attached hydrogens, aromaticity, and ring membership. This scheme yields 22 protein and 70 ligand atom types, forming a feature vector of 1,540 integer-valued descriptors.

Multi-shell occurrence of interatomic contact (MS-OIC)

MS-OIC is a feature generation technique used in OnionNet-2 [29] and OnionNet [39]. Like other methods, it counts occurrences of various protein-ligand entity pairs. For ligands, eight atom types are defined: H, C, N, O, P, S, HAL, and DU, where HAL represents all halogens, and DU includes any atoms not classified in these categories. On the protein side, twenty natural amino acids and an additional "OTH" symbol are considered. This symbol accounts for water molecules, ions, and non-standard amino acids. Residue-atom pair contacts are counted within 62 concentric shells around the ligand, each 0.5 Å thick, providing a detailed representation of the protein-ligand complex. The final feature vector has a dimensionality of 10,416 $(8 \times 21 \times 62)$.

Software architecture

We implement `REINDEER` software with the mindset of minimum dependencies. `REINDEER` operates on standard `Python (v3.9)` libraries, e.g., `Numpy` [40] and `Scipy` [41]. To ensure code quality and consistency, several extensions of `VSCode` software, e.g., `black` and `pylint`, are utilized during development. The exact list of libraries and quality control tools used alongside installation steps are available at the `REINDEER` repository in GitHub. For feature generation, `REINDEER` provides three levels: CLI, GUI, and Python API (see Fig. 2 and Fig. S1 for GUI). It is necessary to mention that ligand and protein structures (assumed to have hydrogen atoms) should be provided in separate MOL2 (SDF) and PDB formats, respectively. Users are expected to add hydrogen atoms, at the desired pH, using external tools such as `PDBFixer` [42] for proteins and `OpenBabel` [43] for ligands. Distance cutoffs for each method are set to their original published values by default (OIC and DWIC: 12 Å, ECIF: 6 Å, MS-OIC: 31 Å with 0.5 Å shells). However, advanced users can modify these parameters via the

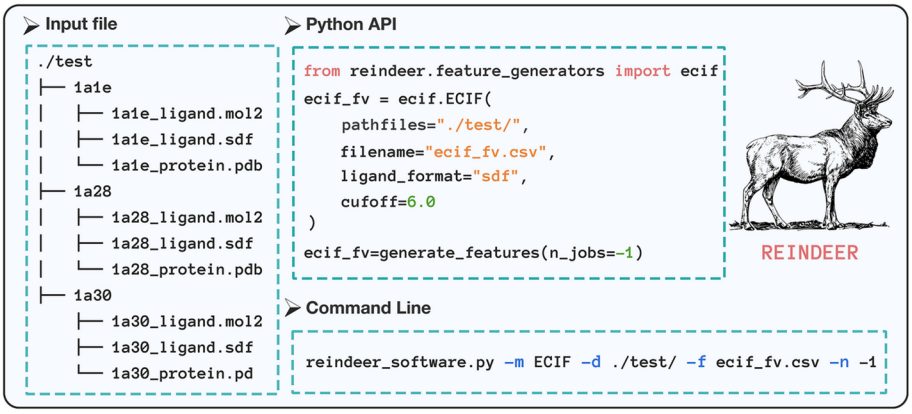

Fig. 2 Schematic of input file, Python API (Application Programming Interface), and Command Line interface for REINDEER package

Python API and CLI to suit dataset-specific requirements. Generated features are saved in comma-separated value (CSV) format in the user-specified directory. The Streamlit package (v.1.25.0) is employed for the REINDEER GUI development (see Fig. S2). Streamlit is an open-source library in Python that simplifies the creation and sharing of custom web apps specifically for machine learning and data science. Instructions on REINDEER usage at CLI and GUI levels or import as a module are available in the REINDEER GitHub repository.

It should be mentioned that REINDEER's implementations of ECIF and MS-OIC are refactored from their original open-source codebases, preserving core algorithmic logic while adding parallelization and unified interface support. Execution times reported here reflect performance within the REINDEER framework and are not intended as direct comparisons against the original standalone implementations.

Case study

We demonstrate the effectiveness of REINDEER by designing an ML-based scoring function for a newly curated protein-peptide dataset. In this study, we utilize XGBoost (eXtreme Gradient Boosting) [34] as our machine learning algorithm because of its proven and reliable performance in cheminformatics and bioinformatics tasks, such as predicting molecular properties and estimating protein-ligand binding affinity [44]. We do not include a comparative analysis with other machine learning algorithms or other state-of-the-art models, as it falls outside the scope of this paper. Algorithms, because they go beyond the scope of the current paper. Additionally, we enhance the scoring function to provide confidence levels for predicted binding affinities by implementing a conformal predictor. In the following sections, we discuss the XGBoost algorithm and conformal prediction mechanism and provide further details on hyperparameters optimization.

XGBoost

A tree-boosting system, XGBoost [34], is selected to develop an ML-based scoring function. XGBoost is an ensemble method that builds models sequentially, each correcting the errors of its predecessor. Unlike the commonly used Gradient Boosting Decision Trees (GBDT) implementation in scikit-learn [45], XGBoost employs Newton's

method—a second-order gradient approximation—to enhance convergence and accuracy. It also incorporates regularization terms to prevent overfitting and efficiently handles large datasets. These features make XGBoost a widely used learning algorithm in bioinformatics and chemistry [46, 47].

Here, we implement and optimize our estimator and its hyperparameters using the `XGBoost` package (v1.7.6) and `Optuna` (v3.6.1). Table S1 in the Supplementary Information (SI) (Additional file 1) summarizes key hyperparameters and their corresponding search space.

Conformal prediction

It is a standard practice to assess an ML performance using external test sets or cross-validation. However, the lack of reported confidence for a new example makes trusting the predictive value unsafe [37]. To address this problem, we opt for an off-line inductive conformal prediction (ICP) framework [36, 37]. The conformal prediction (CP) approach relies on a confidence predictor, a prediction algorithm that outputs a prediction region, e.g., a real–valued interval in a regression case. To construct a confidence predictor, we split a dataset into mutually exclusive training, validation, and test sets. Then, we ought to define a nonconformity measure—an essential aspect of ICP measuring the difference between the test sample and the rest of the data—by the following formula:

$$\alpha_i = \frac{|\hat{y}_i - y_i|}{e^{\sigma_i}} \tag{3}$$

where y_i, $\hat{y}_i$, and σ_i are true label, predicted label, and standard deviation of predicted labels by an ensemble of size K for i^{th} data instance, respectively. In the next step, we calculate σ values for all validation data and sort them in ascending order. We select 80^{th} percentile or P_{80} from the sorted list concerning a desired confidence level. Eventually, we obtain a prediction or confidence region (CR) for new data instance by the formula $P_{80} \times e^{\sigma_i}$, in which σ_i is the standard deviation of the prediction labels of K trained models [13].

Results

We explore the capability of `REINDEER` by designing an ML-based scoring function for protein-peptide complexes. It is essential to indicate that the primary objective of this research is to provide software and a proof-of-concept example. We are not seeking to conduct a comprehensive benchmark against state-of-the-art models, datasets, or introduce additional capability like interpretability. Peptide–based drugs are an emerging class of therapeutics—usually consisting of 2 to 50 amino acids—between small molecules and biologics in size, complexity, and functionality. These drugs have higher specificity and affinity and show lower toxicity [48]. Their applicability spans from oncology and metabolic disorders to cardiovascular diseases [49]. Specifically in metabolic disorders, peptides like glucagon–like peptide-1 (GLP-1) analogs, e.g., liraglutide and semaglutide, are broadly used to treat type 2 diabetes and chronic obesity [50, 51].

Here, we employ a subset (2519 complexes) of the protein-peptide dataset of Durant et al. paper [31] (see Zenodo repository for PDB IDs). They compiled this dataset from the PDBBind 2020 version database [52] by preserving complexes with the letters "MER" as the ligand code for peptides. In the current dataset, all binding affinity values are

reported as pK units (i.e., $-log_{10}K$, where K is the equilibrium dissociation or inhibition constant), which are dimensionless. Figure 3 see Table S2 for quantitative version—displays distributions of binding affinity and molecular properties related to drug-likeness or bioavailability, such as molecular weight and number of hydrogen bond acceptors. This figure illustrates how peptide-based drugs deviate from Lipinski's Rule of Five, resulting in poor ADME (absorption, distribution, metabolism, and excretion) and pharmacokinetic characteristics, as noted in previous research on peptide-based drugs [48].

Then, we investigate the execution time of each feature generation method against different numbers of CPU Cores. To this end, we utilize the GUI of `REINDEER` and measure execution time on a common Linux distribution, equipped with an Intel(R) Core(TM) i7-14700KF, and 64 GB RAM. Figure 4a and Table S3 depict improved parallel efficiency by the decrease in execution time with the increase of CPU cores. However, the extent of this improvement varies between the different algorithms. MS-OIC starts with the longest execution time (≈1288 s) but delivers the most drastic drop, particularly from two to four cores, after which the drop slows but remains apparent up to 16. In contrast, OIC and DWIC have relatively the same execution time profiles with limited reduction after four cores, suggesting bottlenecks or parallel execution inefficiency. ECIF takes more execution time than OIC and DWIC initially but benefits more from parallelization, with noticeable declines up to eight cores before leveling off.

The different scaling demonstrates the algorithmic complexity of each method. OIC and DWIC employ simple counting and distance-weighting within 50 and 200

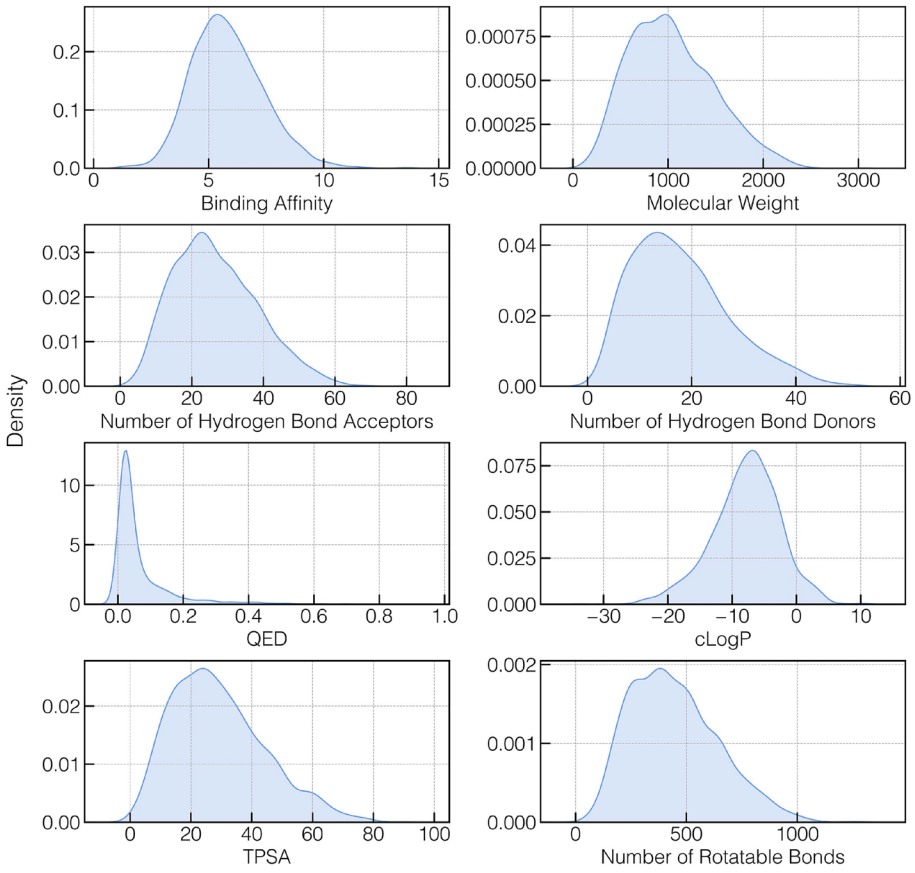

Fig. 3 Binding affinity and molecular property distributions for the protein-peptide dataset

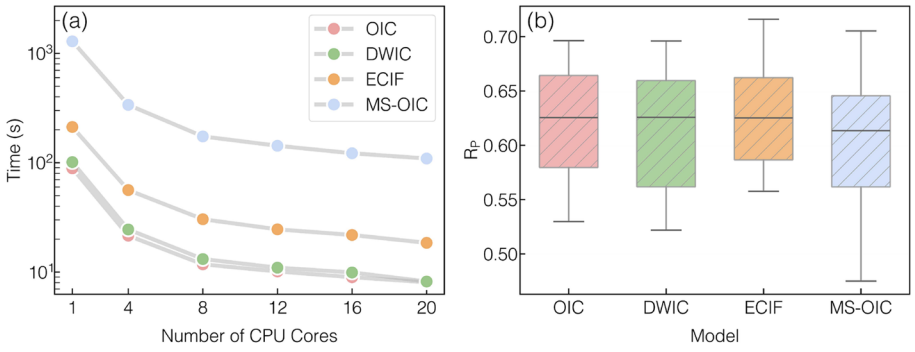

Fig. 4 **a** Execution time vs. number of CPU cores for different algorithms (OIC, DWIC, ECIF, and MS-OIC) and**b** Boxplots diagram of R_P values on the test set

dimensions, respectively. So, their computational cost is dominated by pairwise distance calculations, which parallelize efficiently but can be saturated beyond four cores as parallel overhead outweighs the lightweight per-contact computations. In contrast, ECIF's higher dimensionality (1,540) and complex atom typing shift the burden toward feature accumulation, benefiting from additional cores. MS-OIC, with its 10,416-dimensional output and 62 concentric shells, involves extensive binning operations per atom pair, allowing effective utilization of up to 16 cores before memory bandwidth limits are reached.

Prior to learning, the four generated features (i.e., OIC, DWIC, ECIF, and MS-OIC) are pre-processed by standardizing and discarding static, quasi-static (variance less than 0.01), and correlated features (correlation above 0.95) [28]. This procedure reduces the number of features to 25, 96, 780, and 8604 for OIC, DWIC, ECIF, and MS-OIC, respectively (names of retained features are available on the "additional_files" accompanying this manuscript). We design four ML-based scoring functions for predicting binding affinity by the XGBoost regressor algorithm and four generated features. To this end, we employ 10-fold cross-validation for model selection. In this procedure, we randomly split the training folds into separate training and validation sets, retaining the last fold solely for testing. We minimize computational costs by restricting hyperparameter fine-tuning to the first set of training and validation splits.It should be noted that the employed random 10-fold cross-validation does not control for sequence homology, which may lead to optimistic performance estimates. Table S4 lists the optimized hyperparameters, tuned using `Optuna`'s default settings over 20 trials. Figure 4b shows the spread of Pearson's correlation coefficient (R_P) on the test set (for RMSE see Fig S2 in Additional file 1). Table S5 reports the average and standard deviation of (R_P) and Root Mean Square Error (RMSE) for both validation and test sets across 10 folds.

On the test set, ML scoring functions built with OIC and DWIC features yield the lowest R_P values--0.543 and 0.615, respectively. The ECIF feature vector produces the strongest ML-Scoring function with $\mathrm{R}_P = 0.651$ and RMSE = 1.180. While the MS-OIC-based model performs closely ($\mathrm{R}_P = 0.640$, $\mathrm{R}MSE = 1.196$), we choose the ECIF-based function for its lower computational cost and simpler design–consistent with Occam's razor [53]. It is worth noting that all models exhibited regression toward the mean, with predicted binding affinities concentrated within the middle range, substantially narrower than the full experimental span. This well-documented behavior [31]

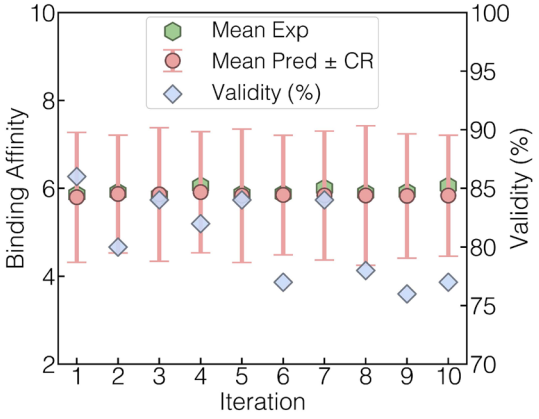

Fig. 5 The mean size of the confidence regions (CR), predictions (Mean Pred), and experimental (Mean Exp) binding affinity values and Validity (in percent) in each set of 10 iterations

likely stems from the underrepresentation of extreme values in the training data and reflects a common challenge in binding affinity prediction.

To test the strength of feature vectors, we build a conformal predictor using the ECIF feature vector and XGBoost to estimate confidence values. Our methodology employs a random train-validation-test split with an 8:1:1 ratio, repeated 10 times to ensure statistical robustness. To boost model diversity, we create an ensemble of $K = 10$ distinct models by additionally sampling 90% of the training splits through bootstrap aggregation.

Figure 5 illustrates the conformal prediction results across these 10 iterations, with detailed numerical results provided in Table S5. The mean predicted binding affinity remains stable across all iterations, with values between 5.8 and 5.9, indicating model robustness. The experimental values show slightly greater variability but generally fall within the confidence regions.

Our conformal predictor achieves remarkably tight confidence regions, with a mean size of almost 1.442 across all repeats. The error bars show little variation, indicating stable uncertainty estimates throughout. The validity metric matches theoretical expectations, averaging 81% across iterations. Individual runs range from 76% to 86%, staying close to the target 80% coverage set by our nonconformity percentile.

The strong match between predicted and experimental values, combined with the appropriate coverage probability, suggests this approach could reliably identify potential false positives in virtual screening applications. Occasional wider gaps—such as in Iteration 4—align with slightly broader confidence regions, showing the model adjusts its uncertainty when needed.

Conclusions

In this work, we introduced `REINDEER`, a tool for generating feature vectors for protein-ligand complexes. `REINDEER` aims to simplify the process for researchers by providing an accessible framework for quickly prototyping ML scoring functions without dealing with complicated implementation details. The current version of `REINDEER` (v0.2.0) comprises several geometry-based feature generation methods, including Occurrence of Interatomic Contact (OIC), Distance-Weighted Interatomic Contact (DWIC), Extended Connectivity Interaction Feature (ECIF), and Multi-Shell Occurrence of Interatomic

Contact (MS-OIC). To enhance computational efficiency, we accelerated the feature generation procedure by leveraging multiprocessor computation, and our analysis indicated that the implemented algorithms scale effectively with the number of CPU cores. Furthermore, we evaluated the performance of the four feature vectors in developing machine learning-based scoring functions for structure-based protein-peptide binding affinity prediction. Our results revealed that the ECIF feature vector, integrated with the XGBoost algorithm, acquired the best performance in terms of R_P and RMSE in a 10-fold cross-validation setting. Additionally, we explored the utility of ECIF and XGBoost in constructing a conformal predictor for uncertainty quantification. By employing a diverse ensemble of 10 models, we demonstrated the efficiency and validity of the approach, which are essential criteria for an effective conformal predictor. While the present study focuses on demonstrating `REINDEER`'s functionality, a comprehensive benchmarking study comparing `REINDEER`-generated features against established peptide-specific scoring and docking methods represents an important direction for future research.

Looking ahead, we plan to expand `REINDEER` in several ways. First, we aim to integrate topology-based feature generation methods, which utilize principles from topology and differential geometry to represent protein-ligand complexes. For instance, techniques such as OPRC-GBT [54], which employs Ricci curvature, and AGL-Score [55], which uses multiscale weighted colored subgraphs, could be implemented to transform protein-ligand complexes into numerical vectors. Additionally, we intend to incorporate Graphics Processing Unit (GPU) support for each algorithm, which could significantly accelerate the feature generation process. While the present study demonstrates the utility of `REINDEER` on a novel protein-peptide dataset, the established CASF-2016 benchmark remains the community standard for evaluating scoring power in protein-ligand binding affinity prediction. A formal evaluation using the CASF-2016 benchmark is planned as a follow-up study.

We believe that `REINDEER`, with its GUI, CLI, and Python integration, provides a user-friendly platform for researchers with minimal programming expertise. Also, it can be a valuable tool for designing machine learning-based scoring functions tailored to specific protein targets. `REINDEER` is freely available under the MIT license and can be accessed via its GitHub repository.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s12859-026-06452-w.

Supplementary file 1 (PDF 620 kb)

Acknowledgements

The authors acknowledge the support and resources from the Center for High-Performance Computing (SARMAD) at Shahid Beheshti University of Iran. This work is based upon research funded by Iran National Science Foundation (INSF) under project No.4037187.

Author contributions

MR and SSN conceptualized the study, MR conducted the experiments, MR and SSN wrote the manuscript. All authors read the manuscript, provided feedback, and eventually approved it in its final form.

Funding

This work is based upon research funded by Iran National Science Foundation (INSF) under project No.4037187.

Data availability
REINDEER software is accessible through https://github.com/miladrayka/reindeer_software link. Protein-peptide dataset is deposited in https://zenodo.org/records/8410136. All codes to reproduce case study results, figures, and tables are available in the CaseStudy.zip file in the GitHub repository.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare no conflict of interest.

Received: 17 May 2025 / Accepted: 15 April 2026
Published online: 22 April 2026

References

1. Harren T, Gutermuth T, Grebner C, Hessler G, Rarey M. Modern machine-learning for binding affinity estimation of protein–ligand complexes: progress, opportunities, and challenges. WIREs Comput Mol Sci. 2024;14:e1716.
2. Kairys V. *et al.* Recent advances in computational and experimental protein-ligand affinity determination techniques. Expert Opin. Drug Discovery 19:649–670.
3. Wang, H. Prediction of protein–ligand binding affinity via deep learning models. Briefings Bioinf. 2024;25:bbae081.
4. Zhang Y, Li S, Meng K, Sun S. Machine learning for sequence and structure-based protein-ligand interaction prediction. J Chem Inf Model. 2024;64:1456–72.
5. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521:436–44.
6. Corso G, Stark H, Jegelka S, Jaakkola T, Barzilay R. Graph neural networks. Nat Rev Methods Primers. 2024;4:1–13.
7. Luong K-D, Singh A. Application of transformers in cheminformatics. J Chem Inf Model. 2024;64:4392–409.
8. Wu Z, et al. From black boxes to actionable insights: a perspective on explainable artificial intelligence for scientific discovery. J Chem Inf Model. 2023;63:7617–27.
9. Mumuni A, Mumuni F. Data augmentation: a comprehensive survey of modern approaches. Array. 2022;16:100258.
10. Ain QU, Aleksandrova A, Roessler FD, Ballester PJ. Machine-learning scoring functions to improve structure-based binding affinity prediction and virtual screening. WIREs Comput Mol Sci. 2015;5:405–24.
11. Liu J, Wang R. Classification of current scoring functions. J Chem Inf Model. 2015;55:475–82.
12. Sánchez-Cruz N, Medina-Franco JL, Mestres J, Barril X. Extended connectivity interaction features: improving binding affinity prediction through chemical description. Bioinformatics. 2021;37:1376–82.
13. Rayka M, Mirzaei M, Latifi AM. An ensemble-based approach to estimate confidence of predicted protein–ligand binding affinity values. Mol Inf. 2024;43:e202300292.
14. Géron A. Hands-on machine learning with Scikit-learn, Keras, and tensorflow: concepts, tools, and techniques to build intelligent systems (" O'Reilly Media, Inc.", 2022).
15. Ashtawy HM, Mahapatra NR. Descriptor data bank (DDB): A cloud platform for multiperspective modeling of protein-ligand interactions. J Chem Inf Model. 2018;58:134–47.
16. Wójcikowski M, Zielenkiewicz P, Siedlecki P. Open drug discovery toolkit (ODDT): a new open-source player in the drug discovery field. J Cheminf. 2015;7:1–6.
17. Ballester PJ, Mitchell JBO. A machine learning approach to predicting protein–ligand binding affinity with applications to molecular docking. Bioinformatics. 2010;26:1169–75.
18. Wójcikowski M, Kukiełka M, Stepniewska-Dziubinska MM, Siedlecki P. Development of a protein–ligand extended connectivity (PLEC) fingerprint and its application for binding affinity predictions. Bioinformatics. 2019;35:1334–41.
19. Durrant JD, McCammon JA. NNScore 2.0: A neural-network receptor-ligand scoring function. J Chem Inf Model. 2011;51:2897–903.
20. Zhang, X. *et al.* ML-PLIC: a web platform for characterizing protein–ligand interactions and developing machine learning-based scoring functions. Briefings Bioinf. 2023;24:bbad295.
21. Zhang X, et al. ASFP (Artificial Intelligence based scoring function Platform): a web server for the development of customized scoring functions. J Cheminf. 2021;13:1–9.
22. Da C, Kireev D. Structural protein-ligand interaction fingerprints (SPLIF) for structure-based virtual screening: method and benchmark study. J Chem Inf Model. 2014;54:2555–61.
23. Lin Z, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. Science. 2023;379:1123–30.
24. Chithrananda S, Grand G. & Ramsundar, B. ChemBERTa: Large-scale self-supervised pretraining for molecular property prediction. arXiv; 2020.
25. Pitt WR, et al. Real-world applications and experiences of AI/ML Deployment for drug discovery. J Med Chem. 2025;2025:68.
26. Yang C, Chen EA. & Zhang Y. Protein–ligand docking in the machine-learning Era. Molecules 2022;27.
27. Rayka M, Karimi-Jafari MH, Firouzi R. ET-score: Improving protein-ligand binding affinity prediction based on distance-weighted interatomic contact features using extremely randomized trees algorithm. Mol Inf. 2021;40:2060084.
28. Rayka M, Firouzi R. GB-score: Minimally designed machine learning scoring function based on distance-weighted interatomic contact features. Mol Inf. 2023;42:2200135.

29. Wang Z, et al. OnionNet-2: A convolutional neural network model for predicting protein-ligand binding affinity based on residue-atom contacting shells. Front Chem. 2021;9:753002.
30. Wang C, Zhang Y. Improving scoring-docking-screening powers of protein–ligand scoring functions using random forest. J Comput Chem. 2017;38:169–77.
31. Durant G, Boyles F, Birchall K, Marsden B. & Deane CM . Robustly interrogating machine learning-based scoring functions: what are they learning? Bioinformatics 2025;41,btaf040 .
32. Sharma K, Sharma KK, Sharma A, Jain R. Peptide-based drug discovery: current status and recent advances. Drug Discovery Today. 2023;28:103464.
33. Craik DJ, Fairlie DP, Liras S, Price D. The future of peptide-based drugs. Chem Biol Drug Des. 2013;81:136–47.
34. Chen T, Guestrin C. in XGBoost: A scalable tree boosting system 785–794. New York, NY, USA: Association for Computing Machinery; 2016.
35. Akiba T, Sano S, Yanase T, Ohta T, Koyama M. in Optuna: A next-generation hyperparameter optimization framework 2623–2631. New York, NY, USA: Association for Computing Machinery; 2019.
36. Arvidsson McShane S, conformal prediction for cheminformatics modeling, et al. CPSign. J Cheminf. 2024;16:1–17.
37. Norinder U, Carlsson L, Boyer S, Eklund M. Introducing conformal prediction in predictive modeling. a transparent and flexible alternative to applicability domain determination. J Chem Inf Model. 2014;54:1596–603.
38. Eklund M, Norinder U, Boyer S, Carlsson L. The application of conformal prediction to the drug discovery process. Ann Math Artif Intell. 2015;74:117–32.
39. Zheng L, Fan J, Mu Y. OnionNet: a multiple-layer intermolecular-contact-based convolutional neural network for protein-ligand binding affinity prediction. ACS Omega. 2019;4:15956–65.
40. Harris CR, et al. Array programming with NumPy. Nature. 2020;585:357–62.
41. Virtanen P, fundamental algorithms for scientific computing in Python, et al. SciPy 1.0. Nat Methods. 2020;17:261–72.
42. Openmm. pdbfixer (2026). https://github.com/openmm/pdbfixer. [Online; accessed 12. Feb. 2026].
43. O'Boyle NM, et al. Open Babel: An open chemical toolbox. J Cheminf. 2011;3:33.
44. Jiang J. *et al.* Artificial intelligence in bioinformatics: a survey. Briefings Bioinf. 2025;26.
45. Pedregosa F, et al. Scikit-learn: machine learning in Python. J Mach Learn Res. 2011;12:2825–30.
46. Hagg A, Kirschner KN. Open-source machine learning in computational chemistry. J Chem Inf Model. 2023;63:4505–32.
47. Schapin N, Majewski M, Varela-Rial A, Arroniz C, Fabritiis GD. Machine learning small molecule properties in drug discovery. Artif Intell Chem 2023;1:100020.
48. Lamers, C. Overcoming the shortcomings of peptide-based therapeutics. Future Drug Discovery 2022;4.
49. Khalily MP, Soydan M. Peptide-based diagnostic and therapeutic agents: where we are and where we are heading? Chem Biol Drug Des. 2023;101:772–93.
50. Deng Y, Park A, Zhu L, Xie W, Pan CQ. Effect of semaglutide and liraglutide in individuals with obesity or overweight without diabetes: a systematic review. Ther Adv Chronic Dis 2022;13:20406223221108064.
51. Knudsen LB, Lau J. The discovery and development of liraglutide and semaglutide. Front Endocrinol. 2019;10:440904.
52. Su M, et al. Comparative assessment of scoring functions: the CASF-2016 update. J Chem Inf Model. 2019;59:895–913.
53. McFadden J. *Life Is Simple: How Occam's Razor Set Science Free and Shapes the Universe* (Basic Books, 2021). https://books.google.com/books?id=WgMSEAAAQBAJ.
54. Wee J, Xia K. Ollivier persistent ricci curvature-based machine learning for the protein-ligand binding affinity prediction. J Chem Inf Model. 2021;61:1617–26.
55. Nguyen DD, Wei G-W. AGL-Score: algebraic graph learning score for protein-ligand binding scoring, ranking, docking, and screening. J Chem Inf Model. 2019;59:3291–304.

Publisher's Note