RSC Advances

PAPER

View Article Online
View Journal | View Issue

Cite this: *RSC Adv.*, 2023, **13**, 10261

3DProtDTA: a deep learning model for drug-target affinity prediction based on residue-level protein graphs†

Taras Voitsitskyi, *[ac] Roman Stratiichuk,[ad] Ihor Koleiev,[a] Leonid Popryho,[a] Zakhar Ostrovsky,[a] Pavlo Henitsoi,[a] Ivan Khropachov,[a] Volodymyr Vozniak,[a] Roman Zhytar,[a] Diana Nechepurenko,[a] Semen Yesylevskyy,[abc] Alan Nafiiev[a] and Serhii Starosyla[a]

Received 14th January 2023
Accepted 26th March 2023

DOI: 10.1039/d3ra00281k

rsc.li/rsc-advances

Accurate prediction of the drug-target affinity (DTA) *in silico* is of critical importance for modern drug discovery. Computational methods of DTA prediction, applied in the early stages of drug development, are able to speed it up and cut its cost significantly. A wide range of approaches based on machine learning were recently proposed for DTA assessment. The most promising of them are based on deep learning techniques and graph neural networks to encode molecular structures. The recent breakthrough in protein structure prediction made by AlphaFold made an unprecedented amount of proteins without experimentally defined structures accessible for computational DTA prediction. In this work, we propose a new deep learning DTA model 3DProtDTA, which utilises AlphaFold structure predictions in conjunction with the graph representation of proteins. The model is superior to its rivals on common benchmarking datasets and has potential for further improvement.

Introduction

Modern drug discovery remains a painfully slow and expensive process despite all the recent scientific and technological advancements. It usually takes several years and the estimated cost of developing a new drug may run over a billion US dollars.[1] More than 30% of all drugs entering phase II of clinical trials and above 58% of drugs entering phase III fail.[2] It was reported that among 108 new and repurposed drugs, reported as phase II failures between 2008 and 2010, 51% were due to insufficient efficacy.[3] This observation highlighted the need for novel *in silico* techniques that can decrease the failure rate by filtering out compounds with low predicted efficacy in the early stages of the drug discovery pipeline. In this regard, the computational methods that assess drug-target binding affinities (DTA) are of great interest[4] because DTA is generally considered one of the best predictors of resulting drug efficacy. Accurate prediction of the DTA is of critical importance for filtering out inefficient molecules and preventing them from reaching clinical trials, thus a multitude of computational DTA techniques have been developed in recent years.

The most accurate computational estimate of DTA could be obtained from atomistic molecular dynamics simulations (either classical, quantum or hybrid) combined with one of the modern techniques of computing the free energy of ligand binding.[5] However, accuracy comes at the cost of very high computational demands, which makes these methods generally impractical for large-scale virtual screening.

That is why the common method of choice for estimating DTA in modern drug discovery is molecular docking, which provides a reasonable compromise between accuracy and computational efficiency.[6] However, it is generally believed that empirical scoring functions used in molecular docking have already approached the practical limit of accuracy, which is unlikely to be improved without introducing an additional computational burden.

In order to address these drawbacks the classical machine learning (ML) methods for determining DTA were developed. These methods do not depend on computing physical interactions between the target protein and the ligand. They are purely knowledge-based and rely on the idea that similar ligands tend

[a]*Receptor.AI Inc., 20-22 Wenlock Road, London N1 7GU, UK. E-mail: taras270698@gmail.com*

[b]*Institute of Organic Chemistry and Biochemistry, Czech Academy of Sciences, CZ-166 10 Prague 6, Czech Republic*

[c]*Department of Physics of Biological Systems, Institute of Physics of The National Academy of Sciences of Ukraine, Nauky Ave. 46, 03038, Kyiv, Ukraine*

[d]*Department of Biophysics and Medical Informatics, Educational and Scientific Centre "Institute of Biology and Medicine", Taras Shevchenko National University of Kyiv, 64 Volodymyrska Str., 01601 Kyiv, Ukraine*

† Electronic supplementary information (ESI) available: The best model architectures, tuned hyperparameters for the benchmark, detailed data on manual cross-validation experiments, InterPro entries statistics in the datasets, and protein residue-level graphs similarity between AlphaFold and experimental structures. See DOI: https://doi.org/10.1039/d3ra00281k

to interact with similar protein targets, which are encoded into the pre-trained neural networks. Applying such models to any given ligand is blazingly fast, which allows using them for screening the numbers of compounds, which are unreachable for molecular docking or MD simulations. However, even the most successful methods from this category, such as KronRLS[7] and SimBoost,[8] were recently outperformed by more modern deep learning (DL) techniques.

Seminal DL methods of assessing DTA used string representation of the target's amino acid sequence and a simplified linear representation of the ligands using SMILES (molecular-input line-entry system), which were subsequently encoded by 1D convolutional neural networks (CNNs) and/or long-short-term memory (LSTM) blocks.[9,10] The same linear representations were shown to be efficient for DTA prediction in combination with generative adversarial networks (GANs).[11] However, it is obvious that linear representations lead to a huge loss of information and keeping the knowledge about connectivity and 3D arrangements of both the target protein and the ligands could improve the results. Indeed, the introduction of graph neural networks (GNNs), which preserve information about connectivity and the 3D arrangement of atoms, improved the scores of the models on most of the benchmarking DTA datasets.[12]

In contrast to the sequence-based techniques, the performance of the GNNs depends strongly on the availability and accuracy of 3D structures of target proteins. The scarcity of such structures limited the size of the training datasets and hampered the progress of GNN-based DTA models. The huge success of AlphaFold[13] in protein structure prediction opens new opportunities for developing better DTA prediction models. Accurately predicted 3D structure of druggable protein domains that don't have experimentally resolved structures, adds unprecedented amounts of data for training ML models and improving their performance.

In this article, we proposed a new method of constructing efficient residue-level protein graphs based on the target's 3D structure predicted by AlphaFold and selected the best GNN architectures for this kind of data. This resulted in a new deep-learning model for predicting drug-target affinities: 3DProtDTA. When applied to common benchmark datasets our model is superior to its rivals on all evaluated metrics. The perspectives of applications and further improvements of our model are discussed.

Materials and methods

Datasets

We evaluated our approach on two widespread benchmark datasets for DTA prediction: Davis[14] and KIBA.[15]

The Davis dataset contains the pairs of kinase proteins and their respective inhibitors with experimentally determined dissociation constant (K_d) values. K_d values were transformed by eqn (1) and transformed scores were used as labels for benchmarking in the same way as in the baseline approaches. There are 442 proteins and 68 ligands in this dataset.

$$pK_d = -\log_{10}\left(\frac{K_d}{1e9}\right) \quad (1)$$

The KIBA dataset comprises scores originating from an approach called KIBA, in which inhibitor bioactivities from different sources such as K_i, K_d and IC_{50} are combined. The KIBA scores were pre-processed by the SimBoost algorithm[8] and the final values were used as labels for model training. Initially, the KIBA dataset contained 467 proteins and 52 498 ligands. For benchmarking purposes, the same authors[8] filtered the dataset to keep only the drugs and targets with at least 10 samples resulting in 229 unique proteins and 2111 unique ligands.

The numbers of affinity scores and unique entries in the datasets are summarised in Table 1.

We used isomorphic SMILES strings for both datasets and UniProt[16] accession codes for the KIBA dataset, provided by DeepDTA,[10] as initial entries representation. For the Davis

Table 1 Summary of the benchmark datasets

Dataset	Proteins	Ligands	Samples
Davis	442	68	30 056
KIBA	229	2111	118 254

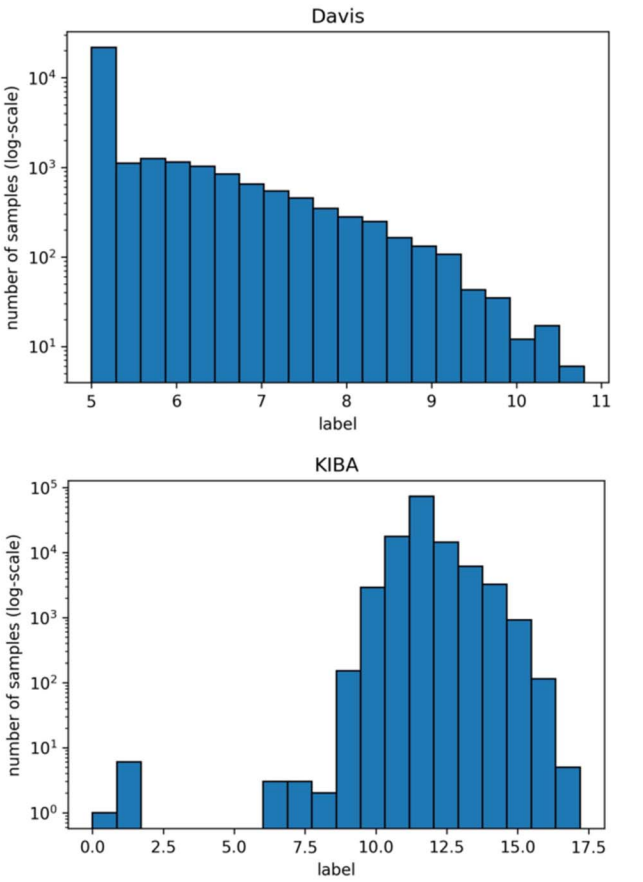

Fig. 1 Distribution of Davis (top) and KIBA (bottom) labels used directly in benchmarking (note that the y-axis is log-scale).

dataset, proteins with the same UniProt accession codes may represent different entries. Therefore, unique identifiers were assigned to each target entry annotated by UniProt accession code, mutation type, phosphorylation status, proteins in a complex, and protein domains. Fig. 1 shows distributions of labels for Davis (a) and KIBA (b) datasets.

Ligand representation

We utilised modified molecular graphs, initially proposed in the approach for drug property prediction Chemi-Net[17] along with the standard Morgan fingerprints[18] to represent ligands for DTA prediction.

Python API of an open-source cheminformatics package RDKit v. 2021.03 was used to generate both ligand representations based on isomorphic SMILES. We calculated 1024 bit vectors of classical Morgan fingerprints with radius 2 and 1024-bit vectors of feature-based fingerprint invariants[19] with the same radius. Both vectors were concatenated into a single 2048-bit vector.

The graphs for ligands were generated on the atomistic level (one node in the graph is one heavy atom in a ligand). Fig. 2 shows distributions of the number of heavy atoms in the used ligands, while Table 2 shows the features for a molecular graph representation of the ligands.

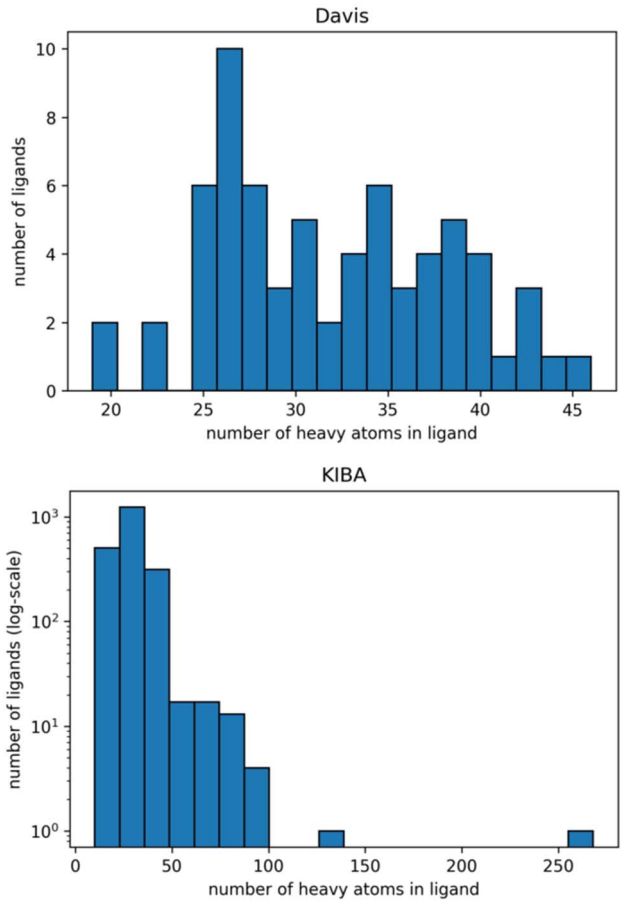

Fig. 2 Distribution of the number of heavy atoms in the ligands in Davis (top) and KIBA (bottom) datasets. The *Y* axis is log-transformed for the KIBA dataset.

Table 2 Atomic-level graph features for ligand representation

Name	Size
Node features	
One-hot encoded atom type (C, N, O, F, S, Cl, Br, P, I)	9
Atom mass (scaled by min–max)	1
Number of directly bonded atom neighbours (scaled by min–max)	1
Total number of bonded hydrogens (scaled by min–max)	1
One-hot encoded atom hybridization (sp^2, sp^3)	2
Is atom in a ring (1 – yes, 0 – no)	1
Is atom aromatic (1 – yes, 0 – no)	1
Is atom hydrophobic (1 – yes, 0 – no)	1
Is atom metal (1 – yes, 0 – no)	1
Is atom halogen (1 – yes, 0 – no)	1
Is atom donor (1 – yes, 0 – no)	1
Is atom acceptor (1 – yes, 0 – no)	1
Is atom positively charged (1 – yes, 0 – no)	1
Is atom negatively charged (1 – yes, 0 – no)	1
Edge features	
One-hot encoded bond type (single, double, triple, aromatic)	4
Bond is conjugated (1 – yes, 0 – no)	1
Bond is in the ring (1 – yes, 0 – no)	1

The Morgan fingerprints and ligand graphs are available in the GitHub repository. The order of the features in the graphs is the same as in Table 2.

Protein representation

We have developed the residue-level protein graph based on 3D protein structures generated by AlphaFold.[13] Approximately 50% of the proteins in both datasets have known 3D structures deposited in the Protein Data Bank but we decided to use AlphaFold predictions for all proteins to make our approach unified and to avoid additional tedious pre-processing of experimentally determined structures, which are often incomplete, contain irrelevant crystallographic ligands, *etc.*

For the KIBA dataset, all the structures were obtained from the AlphaFold protein structure database (**https://alphafold.ebi.ac.uk/**). For the Davis dataset only single protein entries without mutations were downloaded from the AlphaFold database. For the rest of the entries, 3D structures were generated manually using an accelerated implementation of the AlphaFold algorithm in Google Colaboratory: ColabFold.[20]

To avoid undesirable noise from the parts of proteins, which have weak or no relation to the ligand binding, we have parsed domain annotations from UniProt[16] to determine the ligand binding sites. Both datasets contain only the kinase enzymes and the ligands with kinase inhibitor activity. Consequently, only the domains with known kinase activity or related to the kinase activity (annotated by UniProt as a protein kinase, histidine kinase, PI3K/PI4K, PIPK, AGC-kinase, or CBS) were kept in the protein structures.

This preprocessing step not only decreased the noise in the data but also eliminated most residues with a low per-residue confidence score (pLDDT). In AlphaFold a pLDDT is

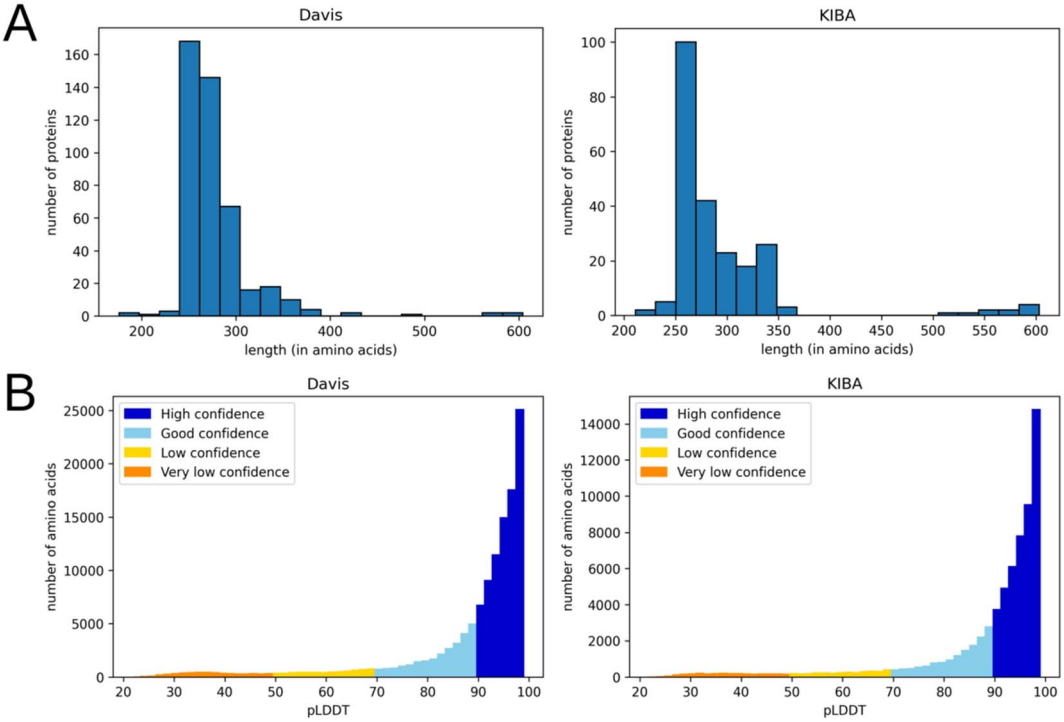

Fig. 3 Distribution of the number of amino acids (A) and pLDDT scores (B) in processed 3D protein structures (after removal of the parts not related to kinase activity) for Davis (left) and KIBA (right) datasets.

a continuous scale from 0–100 (higher is better), which shows the quality of structure prediction. pLDDT lower than 70 emerges in predicted 3D structures if they are unstructured in physiological conditions or the amino acid sequence has low alignment depth13. The regions with such low scores should be treated with caution. Since the domains related to kinase activity are mostly well studied and available in databases of experimental protein structures, keeping only them and removing other regions improves the average pLDDT score. Fig. 3 shows the number of amino acids in processed PDB files (A) as well as the distribution of pLDDT scores (B).

Filtered 3D structures were converted into the residue-level graphs using Biopython v. 1.79 (ref. 21) and Pteros.[22,23] Inspired by the Open Drug Discovery Toolkit,[24] we have developed an approach for encoding protein properties in the graph edge features. An edge was created if two amino acids form an either covalent bond or a non-covalent contact within a particular distance cutoff. The edge features define the type of this connectivity (Table 3). This technique allowed reducing the size of the protein graph in terms of the number of edges compared to the conventional protein graph generation approaches that define the same distance threshold for all types of residue–residue interactions.

Such a graph is directed because of the hydrogen bonds, salt bridges, and cation–pi interactions, which are non-commutative and require specification of the roles for each of involved residues. For example, in the edge created by the hydrogen bond between nodes a and b, it is important to identify which node is a donor and which one is an acceptor. Thus, edge a–b is assigned edge features that are different from those of edge b–a. Similarly, the salt bridges require distinguishing between cationic residue and anionic residue nodes and the cation–pi interactions – between cationic and aromatic residues. In contrast, covalent bonds, pi-stacking and hydrophobic contacts are symmetric.

For each of the 20 standard amino acid types, we assigned seven characteristics AAPHY7 (ref. 25) and 23 BLOSUM62 values according to alignments of homologous protein sequences[26] provided by GraphSol.[27] In addition, the structure-dependent and sequence-dependent node and edge features were used (Table 4).

As the two last node features for the protein graphs we added phosphorylation status and mutation status. Each of the two features is either 1 (phosphorylated/mutated) or 0 (non-phosphorylated/non-mutated) and is the same for each node

Table 3 Bond types and corresponding distance cutoffs used for graph generation and assignment of edge features

Bond type	Distance cutoff (Å)
Covalent	—
Hydrophobic contacts	4
Hydrogen bond	3.5
Salt bridge	4
Cation–pi interaction	5
Perpendicular pi-stacking	5
Parallel pi-stacking	5

Table 4 Residue-level node features for protein graph representation

Name	Size
Node features	
Solvent-accessible surface area (scaled by mean and standard deviation)	1
Phi angle (in degrees, divided by 180)	1
Psi angle (in degrees, divided by 180)	1
One-hot encoded belonging to secondary structure (alpha helix, isolated beta-bridge residue, strand, 3–10 helix, turn, bend)	6
AAPHY7 descriptors of a residue	7
BLOSUM62 descriptors of a residue	23
Phosphorylated (1 – yes, 0 – no, same for all nodes)	1
Mutated (1 – yes, 0 – no, same for all nodes)	1
Edge features	
Covalent bond (1 – yes, 0 – no)	1
Hydrophobic contact (1 – yes, 0 – no)	1
Hydrogen bond (from donor to acceptor; 1 – yes, 0 – no)	1
Hydrogen bond (from acceptor to donor; 1 – yes, 0 – no)	1
Salt bridge (from cation to anion; 1 – yes, 0 – no)	1
Salt bridge (from anion to cation; 1 – yes, 0 – no)	1
Pi-cation interaction (from aromatic ring to cation; 1 – yes, 0 – no)	1
Pi-cation interaction (from cation to aromatic ring; 1 – yes, 0 – no)	1
Parallel pi-stacking (1 – yes, 0 – no)	1
Perpendicular pi-stacking (1 – yes, 0 – no)	1

in a protein graph. It is essential in the case of the Davis dataset due to the presence of protein entries with the same sequence but annotated as phosphorylated/non-phosphorylated or wild-type and mutant entries with internal tandem duplication (ITD) mutation that is hard to translate into sequence unambiguously. Ignoring this data would cause the situation when proteins with identical graph representation have different binding affinities to the same ligand.

Generated protein graphs are available in the GitHub repository. The order of features in the graph is the same as in Table 4.

Model architecture

We used GNNs to extract features from the ligand and protein graphs followed by fully connected (FC) neural network layers. The general model architecture is represented in Fig. 4.

We tuned 3DProtDTA for the single GNN type or the combination of several GNN types that provides the best cross-validation results on the benchmark dataset. The following GNN types were considered: Graph Attention Network that fixes the static attention problem (GAT),[28] Graph Convolutional Network (GCN),[29] Graph Isomorphism Network (GIN),[30] Graph Isomorphism Network with incorporated edge features (GINE),[31] and GCN for learning molecular fingerprints (GMF).[32]

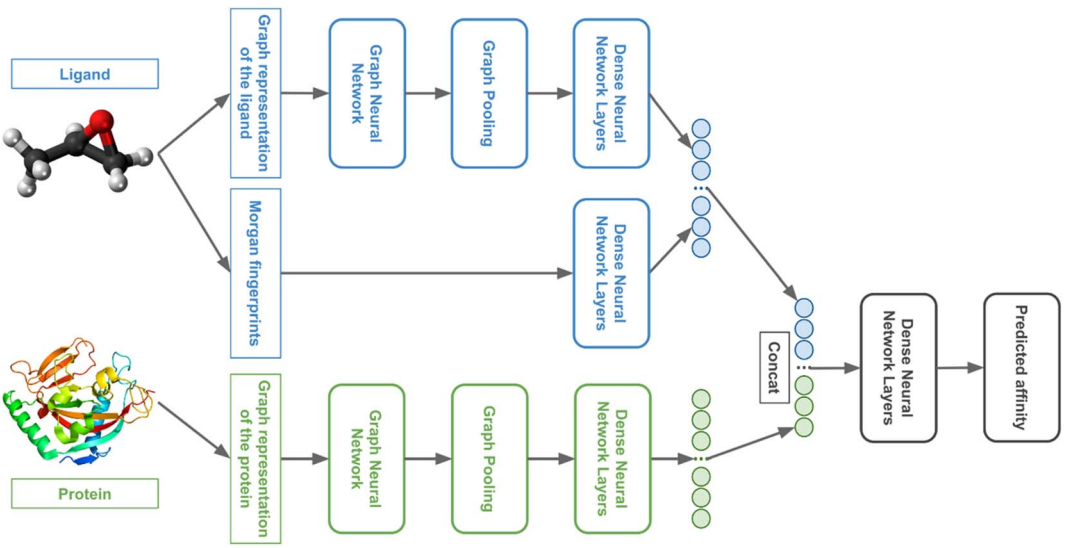

Fig. 4 The general architecture of the 3DProtDTA model.

Besides GNN types, the subjects to tuning included: the configuration of FC layers after 1D input data, GNN pooling output and final dense neural network; dropout rates; activation function for GNN layers; usage of batch normalisation; graph pooling type or combination of types.

The 3DProtDTA model was built with an open-source machine learning framework PyTorch[33] and the GNNs were implemented using PyTorch Geometric.[34]

Comparison with existing techniques

We compared the results of our approach to different classical machine learning-based and deep learning-based methods, which are considered to be state-of-art at the time of writing.

Similarity-based approaches KronRLS[7] and SimBoost[8] used a similarity matrix computed using Pubchem structure clustering server (Pubchem Sim, **https://pubchem.ncbi.nlm.nih.gov**) to represent ligands and the protein similarity matrix constructed with help of Smith–Waterman algorithm to represent targets.[35] KronRLS uses the regularised least-square model while SimBoost is the gradient boosting machine-based method.

The DeepDTA method[10] used 1D CNNs to process protein sequences and SMILES of the ligands. The GANsDTA[11] proposed a semi-supervised GANs-based method to predict binding affinity using target sequences and ligand SMILES. The same initial protein and ligand representations were used in the DeepCDA[9] method, where authors applied encoding by CNN and LSTM blocks. The GraphDTA[12] authors proposed GNNs to process ligand graphs, while proteins were still encoded by CNN applied to sequences.

Evaluation metrics

We selected 3 evaluation metrics used by most authors of the baseline approaches.

The mean squared error (MSE):

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - p_i)^2 \quad (2)$$

where n is the number of samples, y_i is the observed value, and p_i is the predicted value.

The concordance index (CI):[36]

$$\mathrm{CI} = \frac{1}{Z}\sum_{\delta_i > \delta_j} h(b_i - b_j) \quad (3)$$

where b_i is the prediction for the larger affinity δ_i, b_j is the prediction value for the smaller affinity δ_j, Z is a normalisation constant, $h(x)$ is the step function:

$$h(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0.5, & \text{if } x = 0 \\ 0, & \text{if } x < 0 \end{cases} \quad (4)$$

The r_m^2 index:[37]

$$r_m^2 = r^2 \times \left(1 - \sqrt{r^2 - r_0^2}\right) \quad (5)$$

where r^2 and r_0^2 are the squared correlation coefficients with and without intercept respectively.

Experimental setup

The experimental workflow can be split into the following steps: (1) tuning of the best model architecture to use in the benchmarking; (2) benchmarking itself (comparison with other approaches); (3) manual cross-validation experiments to assess the impact of particular choices in the input features model architecture; (4) assessment of the approach performance on cold (unseen by the model) protein target and generalisation over different kinase types.

To assess our approach properly, we used the same train-test and cross-validation data split as in the baseline approaches. Specifically, KIBA[15] and Davis[14] datasets were divided into six equal parts in which one part is selected as the independent test set. The remaining parts of the dataset were used to determine the hyper-parameters *via* 5-fold cross-validation. All 5-fold training sets were used for model training. Subsequently, 5 trained models were applied to predict test set affinity. Finally, an average for each metric was calculated and compared to the baseline approaches. Train and test folds of the datasets were obtained from DeepDTA[10] GitHub repository.

We tuned 3DProtDTA to choose the best GNN type or GNN types combination; the number of multi-head-attentions/size of output sample (number of output node features) in GNNs; usage of activation function after a GNN layer and type of the function (ReLU, Leaky ReLU, sigmoid); the configuration of FC layers; dropout rates; usage of batch normalisation; graph pooling type/types.

The tuning was performed with help of hyperparameter optimization software Optuna v. 3.0.3 (ref. 38) using the tree-

Table 5 The average MSE, CI, and r_m^2 scores of the test set trained on five different training sets for the Davis dataset

Approach	MSE	CI	r_m^2
KronRLS	0.379	0.871	0.407
SimBoost	0.282	0.872	0.644
DeepDTA	0.261	0.878	0.63
GANsDTA	0.276	0.881	0.653
DeepCDA	0.248	0.891	0.649
GraphDTA	0.229	0.893	0.63
3DProtDTA	**0.184**	**0.917**	**0.722**

Table 6 The average MSE, CI, and r_m^2 scores of the test set trained on five different training sets for the KIBA dataset

Approach	MSE	CI	r_m^2
KronRLS	0.411	0.782	0.342
SimBoost	0.222	0.836	0.629
DeepDTA	0.194	0.863	0.673
GANsDTA	0.224	0.866	0.675
DeepCDA	0.176	0.889	0.682
GraphDTA	0.139	0.891	0.673
3DProtDTA	**0.138**	**0.893**	**0.784**

structured Parzen estimator algorithm. We trained the model for 700 epochs and used a batch size of 32, the Adam optimiser with a learning rate of 0.0001, and the mean squared error loss function. The objective of the tuning was to minimise the MSE metric. The other metrics were used for testing the model performance but were not optimised during the training.

After the tuning by the tree-structured Parzen estimator algorithm, we ran a range of manual cross-validation experiments (after obtaining benchmarking results) keeping all the components in the best tuned architecture fixed except:

- The type of GNN architecture for a protein;
- The type of GNN architecture for a ligand;
- The type of graph pooling for both protein and ligand;
- Usage of ligand graph or Morgan fingerprint as the only ligand features.

These experiments were performed in order to assess in depth the impact of GNN architecture, graph pooling and ligand features. While changing the type of GNN model, we kept the size of output (or the number of attention heads in the case of GAT) equal or as close as possible to the tuned parameters.

The last set of experiments was conducted to assess how the 3DProtDTA performs on cold protein targets and, at the same time, to generalise over different kinase types. For that purpose, we annotated the proteins in both datasets using InterPro database entries.[39,40] The number of proteins annotated by the top 20 entries can be found in Tables S6 and S7.† As expected, two major kinase groups in both datasets were tyrosine-protein kinases (InterPro entry IPR008266) and serine/threonine-protein kinases (InterPro entry IPR008271). Subsequently, we randomly selected 10 tyrosine kinases and 10 serine/threonine kinases from each benchmarking dataset and separated all the samples containing those proteins into cold target test datasets. The rest of the samples were filtered to create 3 training datasets (separately for Davis and KIBA benchmarks): (1) all the remaining samples without filtering; (2) the samples with all tyrosine kinases excluded; (3) the samples with all serine/threonine kinases excluded. Consequently, we obtained 3 models per benchmarking dataset: trained on all data, trained on the data without tyrosine kinases, and trained on the data without serine/threonine kinases. We collected the metrics from all cold target test splits.

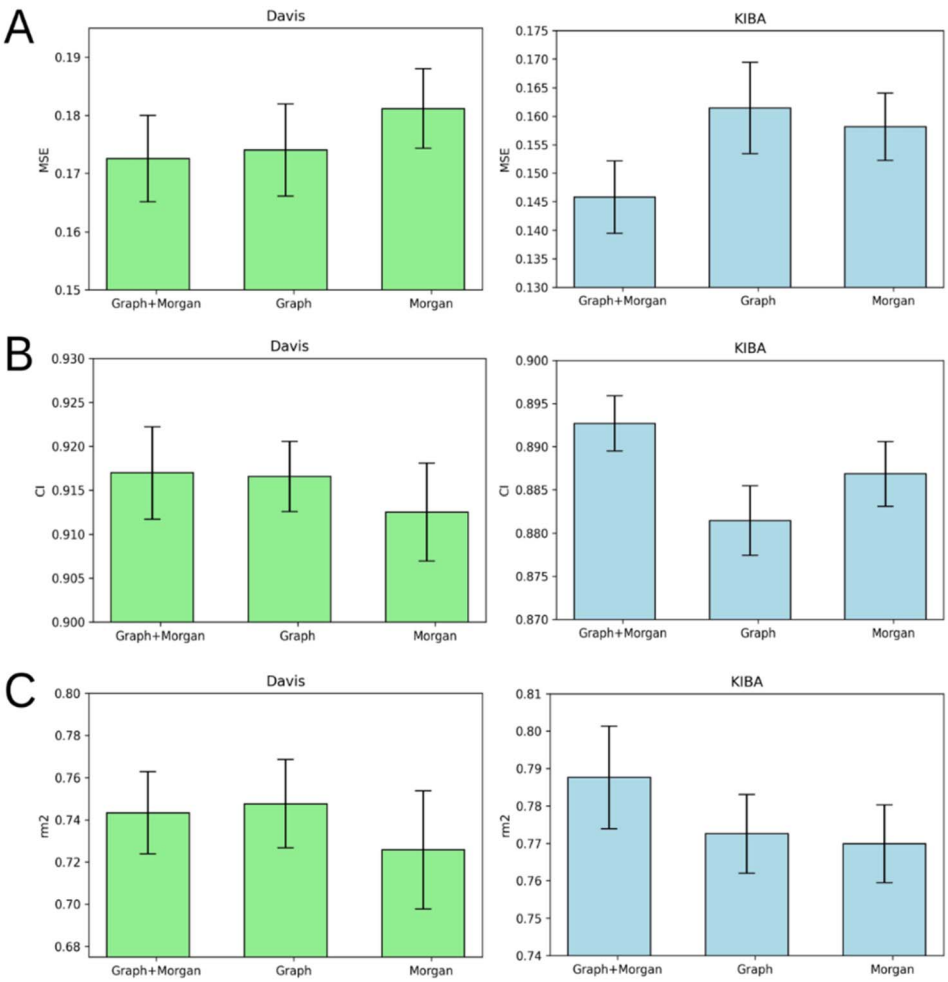

Fig. 5 Average MSE (A), CI (B), and r_m^2 index (C) after 5-fold cross-validation for Davis (left) and KIBA (right) datasets. Models that use molecular graph and Morgan fingerprint (graph + Morgan), only molecular graph (graph) or only Morgan fingerprint (Morgan) as ligand representation are compared. Error bars represent standard deviation.

Results and discussion

Benchmarking

The best model architectures and tuned hyperparameters for the benchmark are available in ESI (Fig. S1†). All the data for model training is freely available in the GitHub repository (**https://github.com/vtarasv/3d-prot-dta.git**).

Tables 5 and 6 compare obtained model performance metrics to the baseline approaches for Davis and KIBA datasets respectively. The best scores are shown in bold.

According to obtained results, 3DProtDTA considerably outperforms other approaches in terms of all 3 metrics on the Davis dataset. In the case of the KIBA dataset, the only metric that demonstrated significant improvement over competitors was r_m^2, while the two other metrics were comparable (but still superior) to the rivals.

Performance of ligand feature types

Atom-level molecular graphs and Morgan fingerprints are widely used types of features in the development of predictive models that take small molecules as input. The result of model training on each type separately or both types together is provided in Fig. 5.

The results demonstrate the nearly equal performance of molecular graph only and combined representation of ligands for the Davis dataset. Nevertheless, the combined representation is clearly superior to others in terms of the KIBA dataset.

We identified the molecular diversity in both datasets defined as 1 – Tanimoto similarity scores based on Morgan fingerprints with radius 3 and averaged for each pair of ligands in the dataset. They are equal to 0.882 and 0.892 for the Davis and KIBA datasets respectively. Despite comparable molecular diversity, the model performance on Davis and KIBA datasets is different when comparing graph only and combined representation. We attribute this to two factors: a different number of ligands (68 in Davis and 2111 in KIBA, Table 1) and a much wider distribution of the number of atoms in the molecules from the KIBA dataset (Fig. 2). The graph only representation is most likely insufficient to generalise over all the molecules in the KIBA dataset, especially, for the small cohort of ligands with more than 50 heavy atoms.

The model trained on the molecular graph and Morgan fingerprint together provided the best average MSE, CI, and r_m^2

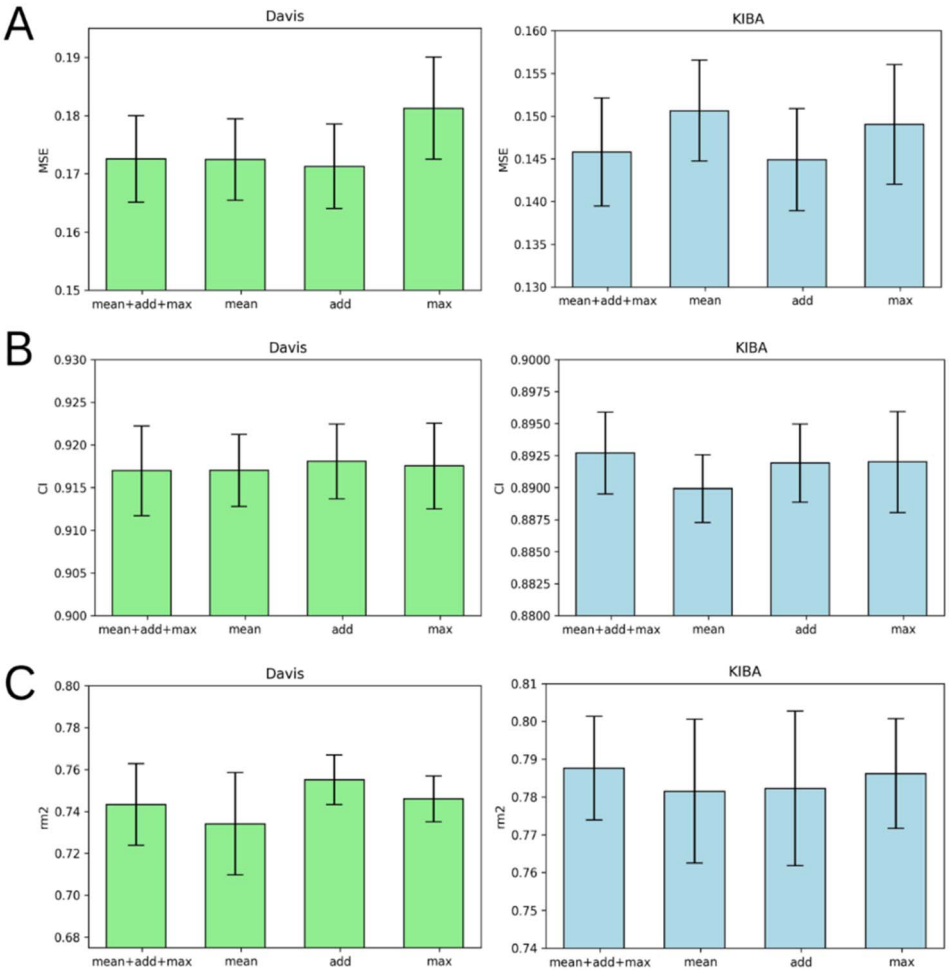

Fig. 6 Average MSE (A), CI (B), and r_m^2 index (C) after 5-fold cross-validation for Davis (left) and KIBA (right) datasets. Models that use mean graph pooling (mean), add graph pooling (add), max graph pooling (max) or all three poolings concatenated (mean + add + max) are compared. Error bars represent standard deviation.

index of both benchmark datasets (Tables S1–S5†). Therefore, we can suggest that these two ligand feature types contain some non-overlapping information useful for DTA prediction.

Performance of graph pooling methods

Graph pooling is a crucial step used to generate the same length 1D latent representation of data processed by GNN for subsequent processing by FC layers. There are three common types of pooling methods including mean pooling, max pooling and add (sum) pooling. Fig. 6 provides a comparison of these pooling approaches.

Fig. 6 demonstrates that add pooling is the best choice for the Davis dataset, while add pooling is comparable to the combined approach (mean + add + max poolings concatenated) in the KIBA dataset cross-validation results. The add pooling provides superior average MSE, CI, and r_m^2 index of both benchmark datasets (Tables S1–S5†).

Performance of ligand GNN types

Fig. 7 illustrates that there is no obvious leader or outsider as the GNN for ligand graph processing. The average of the two benchmark datasets is very close for all the GNN types considering any of the three evaluation metrics (Tables S1–S5†). It is important to highlight that the only GNN model that used edge features during the training was GINE. Thus, the specific characteristic of the edge (like covalent bond type) doesn't play a crucial role in affinity prediction.

Performance of protein GNN types

According to Fig. 8, it is hard to extract a particular GNN model that works the best as a protein graph encoder. The GAT, GCN, GIN and GINE provide very similar average results. The GMF model, however, is noticeably worse (Tables S1–S5†). Analogously to the performance of different GNNs on ligand graphs, the only GNN considering edge features of the protein graph GINE didn't show noticeable metrics improvement relative to other GNN types. This suggests that covalent and non-covalent interactions of amino acid residues alone provide enough information for DTA prediction.

Performance on cold protein target and different kinase types

The performance of our model trained on either all kinases, data without tyrosine kinases, or data without serine/

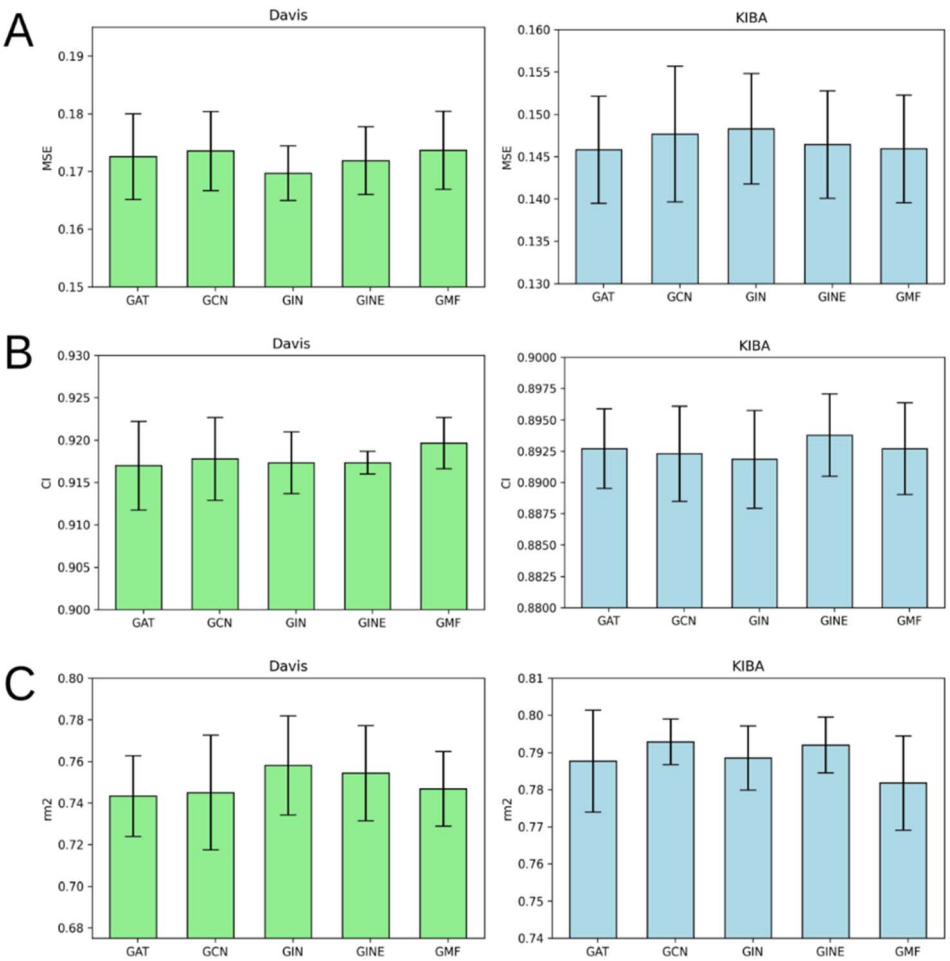

Fig. 7 Average MSE (A), CI (B), and r_m^2 index (C) after 5-fold cross-validation for Davis (left) and KIBA (right) datasets. Models that use GAT, GCN, GIN, GINE or GMF to process ligand atom-level graphs are compared. Error bars represent standard deviation.

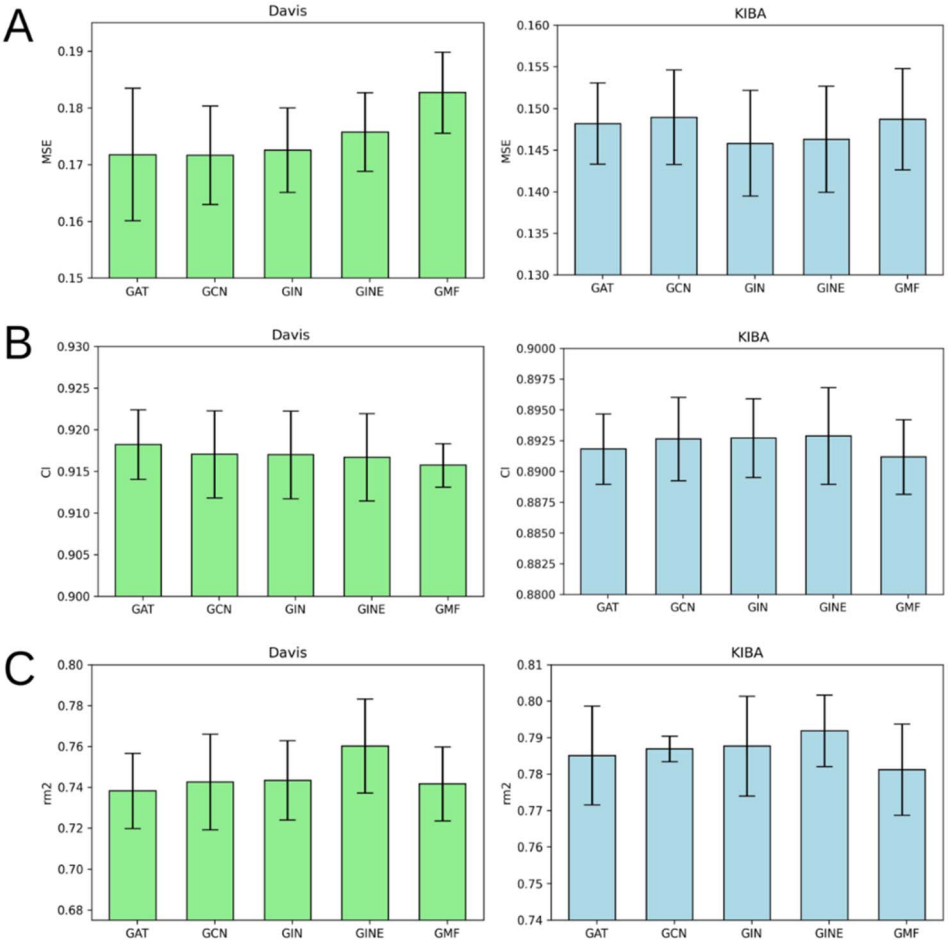

Fig. 8 Average MSE (A), CI (B), and r_m^2 index (C) after 5-fold cross-validation for Davis (left) and KIBA (right) datasets. Models that use GAT, GCN, GIN, GINE or GMF to process protein residue-level graphs are compared. Error bars represent standard deviation.

Table 7 The MSE, CI, and r_m^2 scores of the test sets composed of all samples containing one of 10 random tyrosine or serine/threonine kinases for the Davis dataset

	Test, cold tyrosine kinases			Test, cold serine/threonine kinases		
Train dataset	MSE	CI	r_m^2	MSE	CI	r_m^2
All kinases	0.305	0.880	0.536	0.311	0.889	0.509
No tyrosine kinases	0.719	0.746	0.254	**0.284**	**0.909**	**0.572**
No serine/threonine kinases	**0.239**	**0.909**	**0.629**	0.683	0.682	0.136

Table 8 The MSE, CI, and r_m^2 scores of the test sets composed of all the samples containing one of 10 random tyrosine or serine/threonine kinases for the KIBA dataset

	Test, cold tyrosine kinases			Test, cold serine/threonine kinases		
Train dataset	MSE	CI	r_m^2	MSE	CI	r_m^2
All kinases	0.383	0.668	0.599	**0.236**	0.857	**0.700**
No tyrosine kinases	0.904	0.634	0.155	0.238	**0.858**	0.693
No serine/threonine kinases	**0.359**	**0.729**	**0.625**	0.589	0.713	0.277

threonine kinases was evaluated on a cold protein, as well as on a cold kinase type (Tables 7 and 8). The best scores are shown in bold.

The results show that the model trained on all kinase types performs quite similarly for both tyrosine and serine/threonine kinases in the Davis dataset. In contrast, the model trained on all kinase types of the KIBA dataset demonstrates considerably better performance for serine/threonine kinases. We expected the opposite result because the rate of serine/threonine kinases to tyrosine kinases is roughly 2 : 1 in the KIBA dataset and 3 : 1 in the Davis dataset (Tables S6 and S7†).

The models trained without tyrosine kinases demonstrate the best or nearly the best performance on the cold serine/threonine kinases test and *vice versa*. Their performance is better than for the models trained on all kinase types. This means that adding new kinase types to the dataset may have a negative impact on the model inference for other kinase types. A potential solution to this issue is training a multi-task model, which will contain common layers for all kinase types in the beginning and separate layers for each kinase type at the end. Development and testing of such a model are out of the scope of this work, however.

The models trained without a particular type of kinases perform significantly worse on the test with omitted types of kinases. This is expectable behaviour, which emphasises that the models perform the best for protein types, which were present in the training set.

Limitations and perspectives

Although the usage of protein structures from the AlphaFold database provides data uniformity, it has some drawbacks. Particularly, this approach introduces uncertainty if multiple experimental structures are resolved and used as templates. In such cases, the AlphaFold could potentially produce a blend between several alternative protein conformations, which is not functionally relevant. A potential improvement would be the usage of all available experimentally determined protein structures along with the AlphaFold predictions. However, we observed that the residue-level graphs of several randomly picked experimental structures of kinases are nearly identical to ones generated from the AlphaFold predicted structures (Table S8†). The probable reasons for this are: (1) the abundance of various experimentally determined kinase domain structures available for AlphaFold that positively impacts the quality of predictions (Fig. 3B) and (2) the coarse-grained resolution on the residue-level graph representation used in this work (which would not be the case for atom-level graph representations, though). The usage of the AlphaFold structures provides additional benefits, such as covering proteins without known experimental structures and avoiding problems with incomplete structures and missing or ambiguous atoms.

Another straightforward improvement is the usage of atomic-level protein graphs instead of residue-level ones, and the usage of more comprehensive node and edge features. Particularly, the B-factors and other measures of protein flexibility, such as cross-correlation matrices of motions, could be used.

It is necessary to note that there are other ML-based approaches to predict drug-target affinity, which were not included in this study, such as CSatDTA,[41] FusionDTA,[42] NerLTR-DTA,[43] Masashi's method,[44] WGNN-DTA,[45] *etc.* In these techniques, the experimental setups and data split into training and testing datasets are either different from what we use in the current work or not specified in enough detail. Some of them also utilise specific evaluation metrics, which prevent their direct comparison with other methods.

Our attempts to re-train some of these models using our data split had failed. For example, there is no source code available for NerLTR-DTA, while provided binary crashes. Since the bug reports on GitHub have not been answered for several years, we decided that further attempts to use this technique are futile. The authors of the FusionDTA do not report the amount of computational resources required for the model training. According to our internal estimates, this model is much heavier than 3DProtDTA and its proper re-training is prohibitively long and expensive on our computational facilities. In addition, FusionDTA utilises different hyperparameters for each dataset, which diverges from our concept of the universal model architecture and hyperparameters for all datasets.

Therefore, we decided to limit the scope of our work to the methods, which possess identical setups, datasets and performance metrics and thus allow the apple-to-apple comparison without re-training the third-party ML models.

Conclusions

In this work we developed a new deep learning model for assessing drug–protein affinities called 3DProtDTA. The distinctive feature of this model is graph-based representation of both protein and the ligands, which retain a significant amount of information about their connectivity and spatial arrangement without introducing excessive computational burden. The features for model training were created using the AlphaFold database of predicted protein structures which allows for covering all proteins in two common benchmark datasets. We tuned a wide range of GNN-based model architectures and their combinations to achieve the best model performance. The 3DProtDTA outperforms its competitors on common benchmarking datasets and has a potential for further improvement.

Data availability

https://github.com/vtarasv/3d-prot-dta.git.

Author contributions

SS and AN designed the study and supervised model development and testing. SY coordinated the work, provided scripts for Pteros molecular modelling library and participated in results interpretation. TV developed the modules for features generation, model tuning and testing. TV and RS researched existing methods for drug-target affinity prediction. DN and IK participated in protein preprocessing before feature generation. LP, ZO, and RZ participated in consultations about the best practices and strategies for model tuning and efficient training. PH, IK, and VV performed and supervised the tuning and testing process. The manuscript was written by TV and SY.

Conflicts of interest

All authors are employees of Receptor.AI INC. SS, AN and SY have shares in Receptor.AI INC.

Notes and references

1 A. D. Roses, *Nat. Rev. Drug Discovery*, 2008, **7**, 807–817.
2 G. A. Van Norman, *JACC: Basic Transl. Sci.*, 2019, **4**, 428–437.
3 J. Arrowsmith, *Nat. Rev. Drug Discovery*, 2011, **10**, 328–329.
4 M. Thafar, A. B. Raies, S. Albaradei, M. Essack and V. B. Bajic, *Front. Chem.*, 2019, **7**, 782.
5 L. Pinzi and G. Rastelli, *Int. J. Mol. Sci.*, 2019, **20**, 4331.
6 J. Li, A. Fu and L. Zhang, *Interdiscip. Sci.: Comput. Life Sci.*, 2019, **11**, 320–328.
7 T. Pahikkala, A. Airola, S. Pietila, S. Shakyawar, A. Szwajda, J. Tang and T. Aittokallio, *Briefings Bioinf.*, 2015, **16**, 325–337.

8 T. He, M. Heidemeyer, F. Ban, A. Cherkasov and M. Ester, *J. Cheminf.*, 2017, **9**, 24.
9 K. Abbasi, P. Razzaghi, A. Poso, M. Amanlou, J. B. Ghasemi and A. Masoudi-Nejad, *Bioinformatics*, 2020, **36**, 4633–4642.
10 H. Öztürk, A. Özgür and E. Ozkirimli, *Bioinformatics*, 2018, **34**, i821–i829.
11 L. Zhao, J. Wang, L. Pang, Y. Liu and J. Zhang, *Front. Genet.*, 2020, **10**, 1243.
12 T. Nguyen, H. Le, T. P. Quinn, T. Nguyen, T. D. Le and S. Venkatesh, *Bioinformatics*, 2021, **37**, 1140–1147.
13 J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli and D. Hassabis, *Nature*, 2021, **596**, 583–589.
14 M. I. Davis, J. P. Hunt, S. Herrgard, P. Ciceri, L. M. Wodicka, G. Pallares, M. Hocker, D. K. Treiber and P. P. Zarrinkar, *Nat. Biotechnol.*, 2011, **29**, 1046–1051.
15 J. Tang, A. Szwajda, S. Shakyawar, T. Xu, P. Hintsanen, K. Wennerberg and T. Aittokallio, *J. Chem. Inf. Model.*, 2014, **54**, 735–743.
16 The UniProt Consortium, *Nucleic Acids Res.*, 2019, **47**, D506–D515.
17 K. Liu, X. Sun, L. Jia, J. Ma, H. Xing, J. Wu, H. Gao, Y. Sun, F. Boulnois and J. Fan, *Int. J. Mol. Sci.*, 2019, **20**, 3389.
18 D. Rogers and M. Hahn, *J. Chem. Inf. Model.*, 2010, **50**, 742–754.
19 A. Gobbi and D. Poppinger, *Biotechnol. Bioeng.*, 1998, **61**, 47–54.
20 M. Mirdita, K. Schütze, Y. Moriwaki, L. Heo, S. Ovchinnikov and M. Steinegger, *Nat. Methods*, 2022, **19**, 679–682.
21 P. J. A. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski and M. J. L. de Hoon, *Bioinformatics*, 2009, **25**, 1422–1423.
22 S. O. Yesylevskyy, *J. Comput. Chem.*, 2015, **36**, 1480–1488.
23 S. O. Yesylevskyy, *J. Comput. Chem.*, 2012, **33**, 1632–1636.
24 M. Wójcikowski, P. Zielenkiewicz and P. Siedlecki, *J. Cheminf.*, 2015, **7**, 26.
25 J. Meiler, A. Zeidler, F. Schmäschke and M. Müller, *J. Mol. Model.*, 2001, **7**, 360–369.
26 D. W. Mount, *Cold Spring Harbor Protocols*, 2008, **2008**, pdb.top39.
27 J. Chen, S. Zheng, H. Zhao and Y. Yang, *J. Cheminf.*, 2021, **13**, 7.
28 S. Brody, U. Alon and E. Yahav, *arXiv*, 2021, preprint, arXiv:2105.14491, DOI: **10.48550/ARXIV.2105.14491**.
29 T. N. Kipf and M. Welling, *arXiv*, 2016, preprint, arXiv:1609.02907, DOI: **10.48550/ARXIV.1609.02907**.
30 K. Xu, W. Hu, J. Leskovec and S. Jegelka, *arXiv*, 2018, preprint, arXiv:1810.00826, DOI: **10.48550/ARXIV.1810.00826**.
31 W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande and J. Leskovec, *arXiv*, 2019, preprint, arXiv:1905.12265, DOI: **10.48550/ARXIV.1905.12265**.
32 D. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik and R. P. Adams, *arXiv*, 2015, preprint, arXiv:1509.09292, DOI: **10.48550/ARXIV.1509.09292**.
33 A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala, *arXiv*, 2019, preprint, arXiv:1912.01703, DOI: **10.48550/ARXIV.1912.01703**.
34 M. Fey and J. E. Lenssen, *arXiv*, 2019, preprint, arXiv:1903.02428, DOI: **10.48550/ARXIV.1903.02428**.
35 T. F. Smith and M. S. Waterman, *J. Mol. Biol.*, 1981, **147**, 195–197.
36 M. Gönen and G. Heller, *Biometrika*, 2005, **92**, 965–970.
37 K. Roy, P. Chakraborty, I. Mitra, P. K. Ojha, S. Kar and R. N. Das, *J. Comput. Chem.*, 2013, **34**, 1071–1082.
38 T. Akiba, S. Sano, T. Yanase, T. Ohta and M. Koyama, *arXiv*, 2019, preprint, arXiv:1907.10902, DOI: **10.48550/ARXIV.1907.10902**.
39 M. Blum, H.-Y. Chang, S. Chuguransky, T. Grego, S. Kandasaamy, A. Mitchell, G. Nuka, T. Paysan-Lafosse, M. Qureshi, S. Raj, L. Richardson, G. A. Salazar, L. Williams, P. Bork, A. Bridge, J. Gough, D. H. Haft, I. Letunic, A. Marchler-Bauer, H. Mi, D. A. Natale, M. Necci, C. A. Orengo, A. P. Pandurangan, C. Rivoire, C. J. A. Sigrist, I. Sillitoe, N. Thanki, P. D. Thomas, S. C. E. Tosatto, C. H. Wu, A. Bateman and R. D. Finn, *Nucleic Acids Res.*, 2021, **49**, D344–D354.
40 T. Paysan-Lafosse, M. Blum, S. Chuguransky, T. Grego, B. L. Pinto, G. A. Salazar, M. L. Bileschi, P. Bork, A. Bridge, L. Colwell, J. Gough, D. H. Haft, I. Letunić, A. Marchler-Bauer, H. Mi, D. A. Natale, C. A. Orengo, A. P. Pandurangan, C. Rivoire, C. J. A. Sigrist, I. Sillitoe, N. Thanki, P. D. Thomas, S. C. E. Tosatto, C. H. Wu and A. Bateman, *Nucleic Acids Res.*, 2023, **51**, D418–D427.
41 A. Ghimire, H. Tayara, Z. Xuan and K. T. Chong, *Int. J. Mol. Sci.*, 2022, **23**, 8453.
42 W. Yuan, G. Chen and C. Y.-C. Chen, *Briefings Bioinf.*, 2022, **23**, bbab506.
43 X. Ru, X. Ye, T. Sakurai and Q. Zou, *Bioinformatics*, 2022, **38**, 1964–1971.
44 M. Tsubaki, K. Tomii and J. Sese, *Bioinformatics*, 2019, **35**, 309–318.
45 M. Jiang, S. Wang, S. Zhang, W. Zhou, Y. Zhang and Z. Li, *BMC Genomics*, 2022, **23**, 449.