Hybrid protein-ligand binding residue prediction with protein language models: Does the structure matter?

Hamza Gamouh[1*], Marian Novotny[2], David Hoksza[1*]

[1] Faculty of Mathematics and Physics, Charles University, Malostranské nám. 25, 118 00, Prague, Czech Republic

[2] Faculty of Science, Charles University, Albertov 6, 128 00, Prague, Czech Republic

* hamza.gamouh@matfyz.cuni.cz , david.hoksza@matfyz.cuni.cz

Abstract

Predicting protein-ligand binding sites is crucial in studying protein interactions with applications in biotechnology and drug discovery. Two distinct paradigms have emerged for this purpose: sequence-based methods, which leverage protein sequence information, and structure-based methods, which rely on the three-dimensional (3D) structure of the protein. To enhance the state-of-the-art performance in this field, we propose a novel approach combining both paradigms' strengths. Our hybrid model integrates two recent deep learning architectures: protein language models (pLMs) from the sequence-based paradigm and Graph Neural Networks (GNNs) from the structure-based paradigm. Specifically, we construct a residue-level Graph Attention Network (GAT) model based on the protein's 3D structure that uses pre-trained pLM embeddings as node features. This integration enables our model to capture both the sequential information encoded in the protein sequence and the structural relationships within the protein. The model has improved state-of-the-art performance on a benchmark dataset over a range of ligands and ligand types. Ablation studies have demonstrated the role of the graph attention mechanism, particularly in densely connected graphs. Moreover, we have shown that as more complex pLMs are employed to represent node features, the relative impact of the GNN architecture diminishes. This observation suggests that, to some extent, the structural information required for accurate binding site prediction is inherently captured by the pLMs themselves. protein-ligand binding sites, binding residues prediction, graph neural networks, graph attention, protein language models, protein embeddings

1 Introduction

Proteins are fundamental biomolecules that play a critical role in the functioning of all living organisms. They are involved in various biological processes such as signal transduction or cell regulation and interact with other macromolecules and small molecules to perform their functions. The interaction is mediated through binding sites on the protein surface. These binding sites contain residues crucial for the ligand molecule's recognition and binding. Thus, the study of protein-ligand binding sites and binding residues is essential for understanding the fundamental mechanisms of biological processes with profound impact on applications such as drug discovery [24, 40] and biotechnology [38].

1.1 *Existing work on protein-ligand binding site prediction*

With the rapid advances in computational techniques in the last two decades, various methods have been developed for detecting protein-ligand binding sites. The methods use diverse algorithms and exploit different types of information from protein sequences and 3D structure, broadly categorizing the approaches into sequence-based and structure-based methods [57, 77].

Before describing the existing methods, we should emphasize that the problem of predicting protein-ligand interactions can be approached in two main ways: binding residue prediction, where sequence-based methods are mainly used, and binding site prediction, where structure-based methods are the most appropriate. Binding residue prediction involves labeling individual residues of the protein depending on whether they belong to a binding site. In contrast, binding site prediction aims at detecting surface regions capable of accommodating ligands that can potentially bind to the protein.

Sequence-based methods operate on amino acid sequences and are characterized by their ability to identify binding residues solely from protein sequence data. Although sequence-based methods can only predict individual binding residues and not full binding sites they can still be relevant in many applications, such as variant effect prediction (VEP) as the mutation of a binding residue increases the probability of a detrimental impact of such mutation by hampering the protein's ability to bind ligands [38].

Traditional sequence-based tools, such as ConSurf [9] and S-Site [73], are template-based methods that use proteins with known binding sites as templates together with the evolutionary conservation information to predict binding residues from highly conserved regions of the protein.

In contrast, more recent methods rely on machine learning algorithms to make predictions. With the exponential increase in the size of biological databases [63], there has been an explosion of machine learning methods to solve all kinds of tasks in bioinformatics [59]. In the context of sequence-based methods for protein-ligand binding site prediction, the different machine learning-based methods utilize different types of information about a protein sequence and its amino acids.

Several methods use Support Vector Machines (SVM) and Random Forest (RF) as their main classification algorithms and various input features. TargetS [74] constructs features using evolutionary information from Position Specific Scoring Matrix (PSSM), predicted secondary structure, and ligand-specific binding propensities of residues. ATPint [14] utilizes

evolutionary information, hydrophobicity, and other predicted features such as average accessible surface area. NsitePred [15] computes features from the predicted secondary structure and uses additional information such as the predicted relative solvent accessibility (RSA) and dihedral angles, as well as PSSM features and residue conservation scores. LigandDSES [16] and LigandRFs [17] use amino acid physico-chemcial properties provided by the AAIndex database [36].

Deep learning methods have attracted enormous attention of bioinformaticians in recent years [45] due to their potential of automatic learning of complex representations from vast amounts of available data and due to their recent success in other fields, such as Natural Language Processing (NLP) [37] and Computer Vision (CV) [13]. Deep learning has also been used for binding residue detection in methods such as DeepBind [8] and DeepCSeqSite [19]. These approaches use Convolutional Neural Networks (CNNs) on protein sequences to predict binding residues. DeepBind uses residue types as input features, while DeepCSeqSite relies on various types of information, such as position-specific scoring matrix (PSSM), secondary structure (SS), dihedral angle (DA), and conservation scores (CS).

Recently, language models (LMs) have emerged as a viable option to represent protein sequences. Large LMs have become the standard method in NLP [50] due to their remarkable performance in a wide range of language-related tasks. An example of a very successful LM is the famous ChatGPT, based on the GPT-3 architecture [11], which can generate human-like responses in conversation. In bioinformatics, LMs have also been applied to address various challenges related to protein analysis [47, 64, 79].

A LM is a deep learning model architecture that is trained to learn complex representations of text input, also called embeddings, from an extensive corpus of text. LMs are built upon two basic successful ideas in NLP: masked language modeling and Transformer architecture. Masked language modeling [20] is a self-supervised learning strategy based on masking parts of the text and training the model to predict the missing parts. This strategy benefits from vast amounts of available unannotated data and forces the model to learn general embeddings that can be fine-tuned on downstream tasks where the data is scarce. The Transformer architecture [66] relies on the famous attention mechanism that helps the model attend only to relevant parts of the input by learning the attention weights of different parts of a text input.

Treating protein amino acids as words and sequences as sentences of a natural language opens a way to apply language modeling techniques to proteomics. Recently, several protein language models (pLMs) [25] were constructed by training Transformer architectures on large protein sequence datasets. The learned embeddings of protein sequences were then successfully applied to the prediction of various protein characteristics, such as protein structure [29, 55], or protein-protein interactions [32, 70]. In our recent work, we explored the potential of pLMs to predict protein-ligand binding residues showing superior performance over several state-of-the-art methods on multiple datasets [30].

On the other hand, structure-based methods for protein-ligand binding site prediction utilize features derived from the protein 3D structure. Different structure-based methods vary in the way of representing the 3D protein structure and in the algorithm used for making the predictions.

FINDSITE [12] is a 3D template-based method that uses a threading algorithm based on binding-site similarity to groups of template structures. 3DLigandSite [71] and FunFOLD [56] are also template-based methods that combine sequence and structure similarity to extract homologous proteins from PDB from which ligands are extracted, superimposed, and clustered to determine the binding site associated with each cluster. Various other methods apply geometrical measurements over the 3D structure to detect cavities or hollows on the protein's surface. SURFNET [42] is a method that positions spheres within the space between two protein atoms. LIGSITE [28] detects pockets with a series of simple operations on a cubic

grid. FPocket [44] is based on Voronoi tessellation and alpha spheres. CurPocket [48] defines the binding sites by identifying clusters of concave regions from the curvature distribution of the protein surface. Methods such as Q-SiteFinder [43], FTSite [52] and SiteComp [46] are energy-based methods. Such methods place probes on the protein surface and subsequently locate cavities by estimating the energy potentials between the probes and the cavities. In addition to template-based, geometry-based, and energy-based methods, machine learning methods rely on 3D structural features, sometimes combined with other features, to train various machine learning algorithms. For instance, P2Rank [41] labels solvent-accessible surface points of the protein by using the Random Forest algorithm on a set of handcrafted physico-chemical and structural features. The ligandable points are then clustered to obtain the binding pockets. Recently, deep-learning methods have been introduced for structure-based binding residue/site prediction as well. Often, the methods represent the protein structure as a 3D grid of voxels and use a 3D Convolutional Neural Network (CNN) [53] as their primary model architecture to learn the binding sites. These methods differ mainly in the input features and model hyperparameters. DeepSite [33], PUResNet [35] and DeepSurf [51] employ atomic chemical properties, DeepDrug3D [54] is based on interaction energies of ligand atoms with protein residues, while Deeppocket [7] uses atom types. More recent methods such as SiteRadar [23], GraphPLBR [69], and EquiPocket [76] use different variations of the Graph Neural Network (GNN) architecture and have demonstrated state-of-the-art performance.

GNN is a class of neural networks designed to operate on graphs and other structured data [67]. GNNs are based on the idea of representing the input data as a graph and propagating node information between the graph nodes. Each node is associated with a feature vector containing the node features. These features are iteratively updated by aggregating information from neighboring nodes using a series of message-passing steps. This property of GNNs enables the model to capture the graph's local structure and learn more structure-based and context-aware embeddings. Methods based on GNNs may also benefit from large libraries of predicted protein structures by methods like AlphaFold [34, 65]. The primary output of a GNN is node feature vectors, which can be used for various node-level and graph-level downstream tasks. In recent years GNNs have been applied extensively in bioinformatics and have shown state-of-the-art results across multiple tasks [75].

To our knowledge, no work has been done on bridging sequence-based and structure-based paradigms in the context of protein-ligand binding site prediction. We see this as a promising research avenue as it has the potential to lead to better performance than each of the paradigms separately. In the following sections, we fuse the sequence and structure-based paradigms by building a machine learning model that exploits the two recent state-of-the-art deep learning architectures; a Graph Neural Network augmented with protein-language model embeddings. Particularly, we want to address the following research questions: Can we improve prediction performance by fusing both approaches? How much does the structure information from GNNs contribute to the predictive power of the solely sequence-based pLMs?

2 Methods

The high-level view of our approach, sketched in figure 1, is as follows. The first input of the pipeline is the protein sequence of single-letter amino acid codes. The sequence is processed by a pLM (Embedder), which computes embeddings of each amino acid in the sequence, i.e., residue-level embeddings. The second input is the corresponding protein 3D structure, described as a set of atom 3D coordinates. The structure is converted to a graph by the protein graph constructor (described in ***Protein graph construction***). In the protein graph, nodes correspond to residues labeled by the residue-level embeddings and edges to residues close in the 3D space. The protein graph is then processed by a GNN that predicts

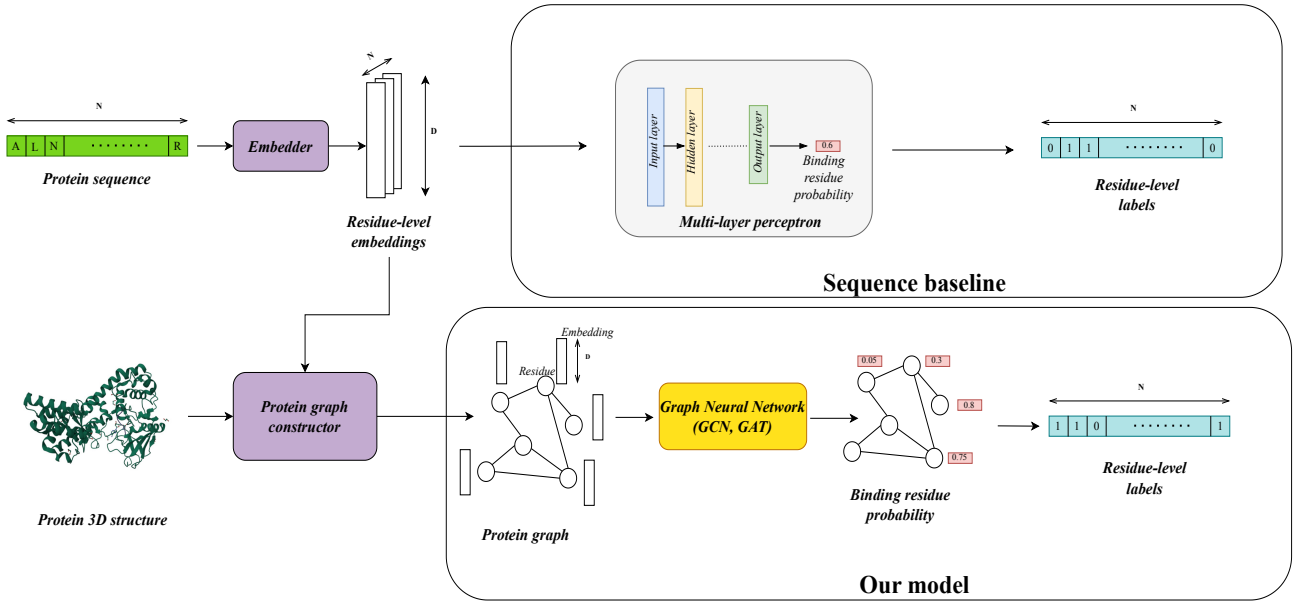


Figure 1. General architecture of our models

binding probabilities for each residue. Using a threshold, the predicted probabilities are converted to binding residue labels (binding vs. non-binding).

Furthermore, we measured the effect of the structure information that comes from the GNN models by comparing them to a baseline model, which is a sequence-based model that lacks graph structure information. The sequence baseline takes the residue-level embeddings as input and feeds it to a multi-layer perceptron which predicts the binding residue probability.

As mentioned, we use the Graph Neural Network (GNN) as our primary model architecture. Different GNN architectures vary in how they aggregate information from other nodes to transform the feature vectors. In our approach, we compare two well-known GNN architectures - Graph Convolutional Network (GCN) [39] and Graph Attention Network (GAT) [68].

The GCN uses convolutional operations to learn feature representations of nodes in a graph. The principle of GCNs is based on the idea of adapting convolutional neural networks (CNNs) [53] to the graph domain by replacing the regular grid-like structure of image data with an irregular graph structure. By analogy, GCNs define a convolution operation on graphs, which involves aggregating information from the node's neighbors and updating the node's feature representation accordingly. The graph convolution works by learning a trainable weight matrix shared across all nodes enabling the GCN to learn a set of filters specific to the graph structure.

The GAT follows the trend of the attention mechanism of the NLP Transformer architectures [66]. The model attends differently to different parts of a given node neighborhood by assigning importance scores to each neighbor based on their relevance to the current node. The attention mechanism enables the GAT to focus on the most relevant nodes in the graph while ignoring noise and irrelevant information. Figure 2 shows the architectural differences between GCNs and GATs.

2.1 *Protein graph construction*

To use the GNN architecture, the protein needs to be represented as a graph with node features. In general, the strength of electrostatic interactions is inversely proportional to

the distances between atoms. Therefore, it is physically plausible to enable information sharing between parts of the protein close to each other. Therefore, to construct the protein graph, we started with the 3D structure of the protein, and we constructed a proximity graph on the residue level. Nodes correspond to residues of the protein, and edges represent the closeness relationship of the residues to each other. Two residues are connected if the distance between their alpha-carbon atoms is less than the threshold distance. In this work, we explored the following thresholds: 4, 6, 8, and 10 Å.

2.2 *Protein language model embeddings*

PLMs process sequences of amino acid letters and return two kinds of embeddings: an embedding for the whole protein sequence and an embedding for each sequence letter, i.e., residue-level embeddings. The latter embeddings can be directly used as node features of the protein graph. We computed the residue-level embeddings with the `bio-embeddings` Python library [2]. In this work, we used two pLMs that are part of the ProtTrans project [22], ProtBERT-BFD that was pre-trained on BFD [61], and ProtT5-XL-UniRef50 (Prot-T5), that was pre-trained on BFD and fine-tuned on UniRef50 [62]. Both language model embeddings are 1024 dimensional.

2.3 *AA Index*

PLMs are context-aware, resulting in different feature vectors for the same amino acid in different sequential contexts. To test the effect of information propagation through the protein graph (see section ***How much does the GNN architecture contribute to the performance?***), we also generated context-independent feature vectors, i.e., vectors whose values are not dependent on the neighborhood, serving as good baseline node features for our GNN models. For that purpose, we used the AAIndex database [36], a large collection of physicochemical and biochemical properties of amino acids. Using the AAIndex database, we constructed node features by collecting all returned properties of the respective amino acid into one vector. We used the Python AAIndex library [1] to extract AAIndex features. The AAIndex features were normalized over all amino acids, resulting in 566-dimensional feature vectors.

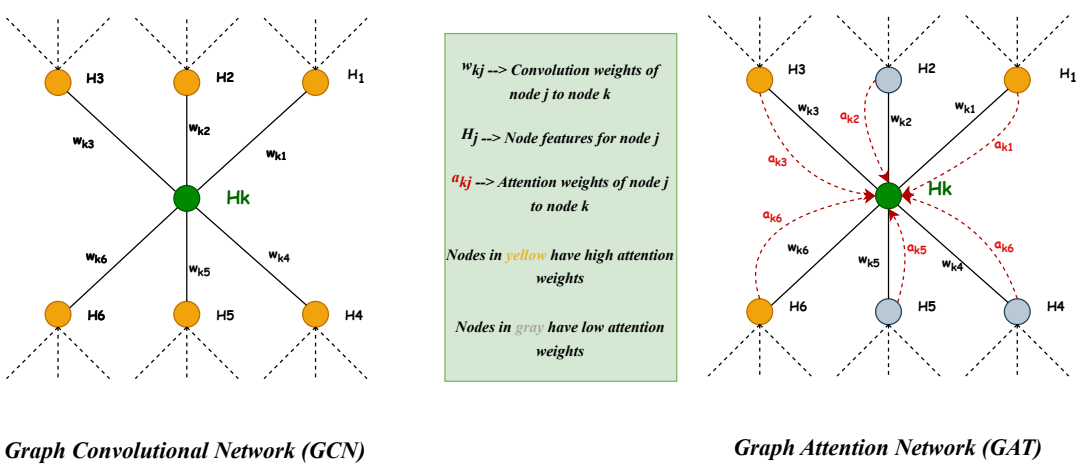


Figure 2. Comparison of GCN and GAT architectures

Ligand	Training sets				Independent test sets		
	Sequences	Missing protein graphs	Binding residues	Non-Binding residues	Sequences	Binding residues	Non-Binding residues
ATP	221	0	3021	72334	50	647	16639
ADP	296	0	3833	98740	47	686	20327
AMP	145	0	1603	44401	33	392	10355
GDP	82	0	1101	26244	14	194	4180
GTP	54	1	745	21205	7	89	1868
Ca^{2+}	965	4	4914	287801	165	785	53779
Zn^{2+}	1168	16	4705	315235	176	744	47851
Mg^{2+}	1138	7	3860	350716	217	852	72002
Mn^{2+}	335	1	1496	112312	58	237	17484
Fe^{3+}	173	1	818	50453	26	120	9092
DNA	335	0	6461	71320	52	973	16225
HEME	206	1	4380	49768	27	580	8630

Table 1. Dataset size.

2.4 *Datasets*

For building and testing our models, we used the benchmark datasets designed by Yu et al. [74]. The benchmark contains training and independent test sets of protein sequences and their corresponding actual binding residues for 12 different ligands, which include: 5 nucleotides (AMP, ADP, ATP, GTP, GDP), 5 ions (CA, MG, MN, FE, ZN), DNA, and HEME.

As the benchmark was used to test several sequence-based methods such as [74] and [30], and given that our method has a structural component, we needed to collect the corresponding 3D structures of the protein sequences. To achieve this, we downloaded the entire BioLip dataset [72], which was used to construct the benchmark, and we extracted the tertiary structures of the sequences by matching their PDB IDs and chain IDs. For sequences whose corresponding structures were not found in BioLip, we used the latest version of PDB [10] to extract the structures.

The PDB files were first parsed by the Biopython library [3] in order to obtain the sequences and the atomic coordinates. Some of the sequences obtained from the Biopython parser underwent minor manual corrections to match them with the sequences from the benchmark dataset. In total, the letters of some modified residues were changed for 12 sequences, one residue was skipped for 13 sequences, and 2 sequences were skipped due to a high mismatch between the sequence retrieved from the benchmark and the sequence retrieved after processing the PDB file. Finally, each residue from a sequence was associated with a 3D coordinate. The obtained coordinates were used to construct the protein graphs as described in section ***Protein graph construction*** using the Python Deep Graph Library (DGL) [4].

We also need to note that due to technical problems with the ProtT5 embeddings, we could not obtain embeddings for all of the proteins. In total, we could not obtain the protein graphs for 31 sequences. The sequences for which we could not generate the embeddings consisted of training sequences only, so this issue did not affect the reported results, as those were based on the test sets. Table 1 illustrates statistics of the benchmark datasets as well as the number of obtained protein graphs after the preprocessing phase.

2.5 *Model hyperparameters*

For building our models, we used the implementation of GCN and GAT provided by the Python library DGL-LifeSci [5]. Our GCN architecture consisted of graph convolutional layers of size 512 with ReLU activation, a dropout rate of 0.5 [60], residual connections [27] and batch normalization [31]. At the same time, our GAT architecture consisted of graph attention layers of size 512, ReLU activations, dropout rate of 0.5, 4 attention heads, and residual connections. We used a dense layer with two softmax units on top of the GCN and GAT models to compute the node-level outputs. We also utilized a weighted version of the binary cross-entropy loss due to the high class imbalance of the datasets, the AdamW optimizer [49] as the optimization algorithm with `learning_rate=3e-4` and `weight_decay=1e-5`, and we trained all the models for `2000 epochs` with a `batch_size=32`. All hyperparameters were chosen after several trials of manual tuning, and all models were trained using the Pytorch Python library [6]. Furthermore, we fixed the architecture of the sequence baseline to a 3-layer Multi-Layer-Perceptron (MLP) with 64 units, ReLU activation, 0.5 Dropout rate, and a final output layer of two units with Softmax activation. These hyperparameters were chosen after several turns of manual experimentation and tuning.

3 Results and Discussion

To evaluate the residue-level predictions of our models, we used standard binary classification metrics. Specifically, we have chosen to show our results with respect to the Matthews Correlation Coefficient (MCC) due to the significant class imbalance present in the datasets, as it has been shown that the MCC metric is one of the most suitable metrics in such cases [18].

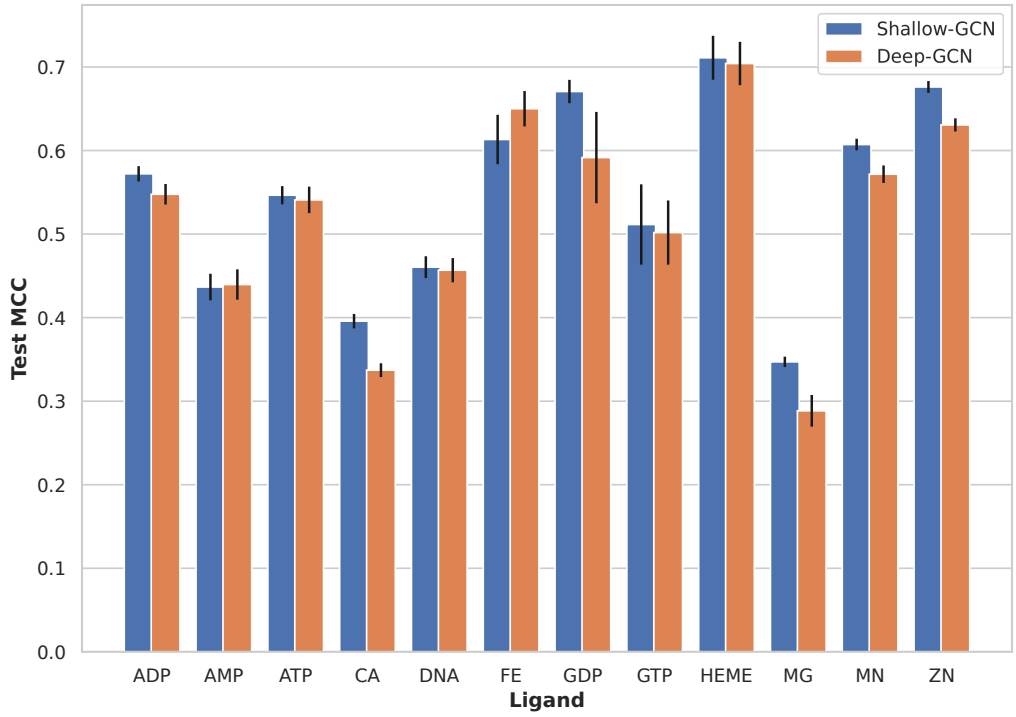
Our recent work [30] shows that more complex LMs often yield better performance. Therefore we used the Prot-T5 embeddings in most of our experiments as one of the most complex pLMs.

We used a random split of the processed benchmark training sets to obtain training and validation sets. The training/validation split ratio was designed for the validation sets to have the same size as the independent test sets. The validation sets were used to define the early stopping epoch while training the models. The training was stopped at the epoch with the best validation MCC. In the subsequent sections, we report the results of the independent test sets.

3.1 *Effect of the number of graph convolutional layers*

The effect of information propagation through the protein graph can best be seen by varying the number of convolutional layers. One round of graph convolution collects information from the neighborhood of a given node. Thus, as the number of graph convolutions increases, a given node will have access to more distant neighbors since the one-hop neighbors will already contain information about farther neighbors in their hidden features computed from previous rounds of graph convolution. Therefore, increasing the number of convolutional layers enables information propagation between distant parts of the graph. To test the effect of the number of convolutional layers on the prediction performance, we used graphs constructed using 6 Å cutoff distance, T5 embeddings, and we compared two versions of our standard GCN architecture, one with a single layer (called `shallow-GCN`) and another one with six layers (called `deep-GCN`). Furthermore, we report the mean and standard deviation of the test score for 5 train/validation random splits to show the effect of different random splits on performance. The results are shown in figure 3 that was created using table 5 of the Supplementary material section.

We can observe that for nearly half of the ligands, both models have very similar



Notes: The bars represent the mean of the test scores for 5 train/val random splits. The error bars represent the standard deviation of the test scores. The *Shallow-GCN* model has one graph convolutional layer with 512 units while the *Deep-GCN* model has six graph convolutional layers with 512 units

Figure 3. Effect of the number of graph convolutional layers.

performance, and the `shallow-GCN` model yields slightly better scores than the `deep-GCN` model in the remaining cases. This suggests that there is little effect of adding more graph convolutional layers.

Moreover, the test scores were quite stable across the different random splits in most cases, especially for the shallow model. Therefore the model's performance is not very sensitive to the chosen training/validation split. Based on the above observations, we stick to the shallow architecture and an arbitrary random split with the same random seed in the remaining experiments. Another reason for choosing shallow GNNs in the following experiments is to avoid the common oversmoothing problem in GNNs [58], where deep GNNs result in nearly indistinguishable node features in the last layers of the network, which may result in a poor performance in downstream tasks.

3.2 *Effect of graph cutoff distances*

Next, we tested the effect of the graph cutoff distance. The cutoff distance influences the graph's connectivity as a higher cutoff distance results in more connections and thus leads to a more densely connected graph. In such a graph, a given node has more neighbors, and

	GCN					GAT					Sequence Baseline
Ligand	4 Å	6 Å	8 Å	10 Å	Ensemble	4 Å	6 Å	8 Å	10 Å	Ensemble	Baseline
ADP	0.569	0.564	0.581	0.557	0.584	0.571	0.578	**0.597**	0.582	0.583	0.542
AMP	0.450	0.412	0.424	0.419	0.445	0.449	0.463	**0.489**	0.475	0.482	0.412
ATP	0.546	0.537	0.538	0.557	0.569	0.566	0.575	0.572	**0.587**	0.583	0.501
CA	0.396	0.382	0.403	0.420	0.421	0.383	0.408	0.408	0.411	**0.426**	0.334
DNA	0.473	0.476	0.470	0.459	0.490	0.460	0.483	**0.510**	0.488	0.499	0.433
FE	0.618	0.645	0.614	0.645	0.645	0.704	0.668	0.692	**0.719**	0.703	0.660
GDP	0.665	0.668	0.737	0.693	0.710	0.696	0.695	**0.746**	0.705	0.744	0.587
GTP	0.537	0.514	0.575	0.564	0.556	0.666	0.669	0.670	0.573	**0.695**	0.516
HEME	0.689	0.672	0.736	0.675	0.691	0.675	0.674	**0.743**	0.682	0.685	0.700
MG	0.343	0.344	0.351	0.362	**0.365**	0.325	0.347	0.364	0.349	0.364	0.297
MN	0.617	0.606	0.594	0.590	0.634	0.602	0.642	0.607	**0.642**	0.638	0.588
ZN	0.660	0.681	0.673	0.693	0.699	0.670	0.672	0.685	0.690	**0.699**	0.655

Table 2. Effect of graph cutoff distance and the graph attention mechanism.

therefore more nodes are taken into account in information propagation to determine the state of the given node. A typical cutoff seen in other works is 6 Å computed based on the distance of alpha carbons [26]. This work tested the following cutoff distances: 4 Å, 6 Å, 8 Å, and 10 Å. Moreover, we constructed an ensemble model using models trained on graphs built using the above cutoff distances. This model combines the predicted binary classes from each cutoff distance and outputs the most often observed class. An ensemble model that uses multiple cutoff distances removes the bias of choosing a predefined cutoff distance. Therefore it has the potential to improve the generalization capability of the GNN.

Table 2, compares the different cutoff distances and the ensemble model. We can observe that although the graph cutoff distance significantly affects the performance of the GCN model, there is no observable consistent trend by varying the cutoff distance. For instance, for some ligands such as ADP and HEME, a low cutoff distance (4 Å) results in higher performance of the GCN, while for other ligands such as CA and MG, a high cutoff distance (10 Å) improves the performance. Moreover, table 6 of the Supplementary material section shows the effect of cutoff distances across multiple classification metrics, namely MCC, together with ROC-AUC, Precision and Recall.

3.3 *Effect of graph attention mechanism*

For GAT, we tested the effect of the graph attention mechanism initially designed as a regularization strategy for the GNN models. The attention may contribute to a better generalization performance as the model attends only to relevant parts of the neighborhood of a node. To test the added value of the graph attention mechanism, we compared our shallow GCN model, with a shallow version of GAT where we used our standard GAT architecture with a single graph attention layer. Table 2 compares the GCN and GAT models for the different cutoff distances.

We see that, unlike in the case of GCN, for most datasets, there is a consistent improvement in the performance of the GAT model with increasing cutoff distance. This observation can be explained by the capacity of the attention mechanism to reduce noise in larger neighborhoods. For graphs obtained using a high cutoff distance, each node has a bigger neighborhood and collects information from more (distant) neighbors. Without using the attention mechanism, the model does not have the capacity to filter out irrelevant information. The graph attention

	AUC				MCC			
Ligand	TargetS	EC-RUS	SXGBsite	T5 GAT Ensemble	TargetS	EC-RUS	SXGBsite	T5 GAT Ensemble
ADP	0.896	0.872	0.907	**0.952**	0.507	0.511	0.521	**0.583**
AMP	0.83	0.815	0.851	**0.903**	0.359	0.393	0.366	**0.482**
ATP	0.898	0.871	0.886	**0.946**	0.502	0.506	0.448	**0.583**
CA	0.767	0.77	0.757	**0.842**	0.243	0.225	0.167	**0.426**
DNA	0.836	0.814	0.827	**0.934**	0.377	0.319	0.27	**0.499**
FE	0.945	0.936	0.913	**0.983**	0.479	0.49	0.454	**0.703**
GDP	0.896	0.872	0.93	**0.971**	0.55	0.579	0.678	**0.744**
GTP	0.855	0.861	0.883	**0.926**	0.617	0.641	0.572	**0.695**
HEME	0.907	**0.935**	0.9	0.876	0.598	0.64	0.555	**0.685**
MG	0.706	0.78	**0.819**	0.783	0.294	0.317	0.326	**0.364**
MN	0.888	0.891	0.888	**0.927**	0.449	0.31	0.329	**0.638**
ZN	0.936	0.958	0.892	**0.966**	0.527	0.437	0.363	**0.699**

Table 3. Comparison with existing methods

mechanism fixes this issue by adjusting the neighbor weights to attend only to neighbors relevant for making the prediction.

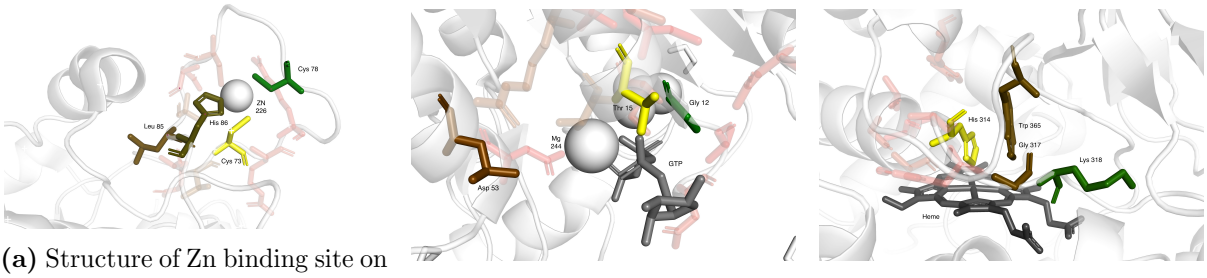
Moreover, we observe that for all ligands, the ensemble models have a very close performance to the best-obtained model via different cutoff distances. The above observations may suggest that the ensemble models can serve as a proxy for the best graph cutoff distance regardless of the ligand type. We can thus compare the different model architectures by comparing their ensemble models. Table 2 shows that the ensemble GAT has significantly higher performance than the ensemble GCN model for the GTP ligand, while it is slightly more performant for most other ligands. Furthermore, the ensemble GAT model significantly outperforms the sequence baseline for most ligands. In the following experiments, we, therefore, consider our best-performing model architecture to be the ensemble GAT model. Table 6 of the Supplementary material section then also includes a comparison of GAT and GCN using more classification metrics.

3.3.1 *What is the attention attentive to?*

In the previous section, we showed that attention helps to improve the accuracy of predictions in comparison with GCN; we were further wondering what amino acids were helpful and therefore investigated a number of binding sites of Zn ion, GTP and HEME as three variable representatives of studied ligands. We were specifically investigating cases where a ligand-binding residue was not predicted by GCN but was correctly predicted with GAT. To do that, we used 10 Å protein graphs; for every binding residue, we extracted the attention value for each neighbor. As our model uses four attention heads, the attention values were averaged across the heads. Then, individually for each binding residue, we colored the binding site with a relative contribution of attention of the binding residue neighbors. We observed that in many cases, the residues with the highest attention were the other ligand-binding residues (and sequence neighbors of the studied residue). In many cases, the binding residues were often physically close to the ligand, but we also observed cases where the residues with the highest attention were on the other part of the binding site and away from the studied residue (see Figure 4).

We should emphasize that the goal of this exercise was to offer a visual way of inspecting the attention, but a more quantitative approach should be taken to draw a conclusive

statement regarding the attention. This is further supported by the fact that we also encountered instances where it was not clear how could the residues with high attention contributed to the accurate prediction of the studied residue.



(a) Structure of Zn binding site on zinc finger antiviral protein (3u9g) – Cys 73 was correctly predicted with the help of attention – the biggest contribution came from Cys 78 (green) and His 86 and Leu 85 (both brown) - Cys 78 and His 86 are directly interacting Zn 226.

(b) Structure of GTP binding site of dethiobiotin synthetase (3qxj) – Thr 15 was correctly predicted with the help of attention – the biggest contribution came from Gly 12 (green) and Asp 53 (brown). Gly 12 is directly interacting with GTP.

(c) Structure of HEME binding site of fungal magkatg2 (3UT2) – His 314 was correctly predicted with the help of attention – the biggest contribution came from Lys 318 (green), Trp 365, and Gly 317 (brown). All three residues are directly interacting with heme.

Figure 4. Visualization of the attention. The binding residue and its neighbors are represented as sticks. The binding residue is colored yellow with neighbors going from green (highest attention) to red (lowest attention). The low-attention neighbors are partially transparent. The ligand is colored gray.

3.4 *Comparison with existing methods*

To put the proposed approach in the context of existing research, we compared our GAT ensemble model with the Prot-T5 embeddings, which consistently demonstrated higher performance in the previous experiments, to other state-of-the-art approaches, namely TargetS [74], EC-RUS [21], and SXGBsite [78]. These methods were trained and tested using the same benchmarking dataset. For each of the three methods, we show the results of the best-performing versions of those methods as presented in the respective papers. Table 3 compares the methods using the area under ROC curve (ROC-AUC) and MCC. Our GAT ensemble model with T5 embeddings outperforms all of the methods on the MCC metric for all ligand datasets, and on the ROC-AUC metric for most datasets. However, it should be emphasized that the presented methods are sequence-based, using only predicted structural features (such as predicted secondary structure). On the other hand, the presented approach does not incorporate 3D structure directly, as the protein graph only approximates the 3D information.

3.5 *How much does the GNN architecture contribute to the performance?*

In the previous sections, we observed that the GNN architecture significantly improves the performance of the ProtT5 pLM. This observation has led us to quantify the contribution of the structure information processed by the GNN architecture to the predictive performance of sequence-based pLMs. To achieve this end, we designed an experiment to compare the

GAT ensemble model with ProtT5 embeddings to two other GAT ensemble models. The first model used a less complex, but still context-aware, pLM ProtBERT for constructing node features. The second model used the context-independent AAIndex physico-chemical properties of amino acids, meaning that a residue is represented by the same feature vector independent of its sequential context. We measured the effect of the structure information by calculating the relative improvement of the GAT ensemble models over their respective sequence baselines. The different test MCC scores and their respective relative improvements are presented in table 4. A more detailed comparison using more classification metrics between the node features is available in table 7 of the Supplementary material section.

	AAIndex			Prot-BERT			Prot-T5		
Ligand	Sequence Baseline	GAT Ensemble	Relative improvement	Sequence Baseline	GAT Ensemble	Relative improvement	Sequence Baseline	GAT Ensemble	Relative improvement
ADP	0.067	0.188	**1.791**	0.310	0.502	0.617	0.542	0.583	0.077
AMP	0.057	0.111	**0.928**	0.255	0.293	0.149	0.412	0.482	0.170
ATP	0.063	0.136	**1.176**	0.326	0.476	0.459	0.501	0.583	0.163
CA	0.115	0.139	0.216	0.191	0.372	**0.943**	0.334	0.426	0.275
DNA	0.121	0.175	**0.445**	0.343	0.393	0.146	0.433	0.499	0.151
FE	0.180	0.279	**0.547**	0.550	0.622	0.132	0.660	0.703	0.065
GDP	0.108	0.249	**1.318**	0.472	0.603	0.276	0.587	0.744	0.267
GTP	0.106	0.235	**1.215**	0.417	0.618	0.485	0.516	0.695	0.346
HEME	0.115	0.206	**0.790**	0.404	0.554	0.371	0.700	0.685	-0.022
MG	0.065	0.098	0.508	0.190	0.333	**0.750**	0.297	0.364	0.227
MN	0.166	0.243	**0.461**	0.420	0.527	0.256	0.588	0.638	0.085
ZN	0.257	0.349	**0.357**	0.506	0.621	0.227	0.655	0.699	0.067

Note: The relative improvement in the MCC score is computed as the GAT ensemble model's MCC score minus the sequence baseline's MCC score, divided by the MCC score of the sequence baseline.

Table 4. Effect of node features.

In the case of the AAIndex node features, we can observe the biggest relative improvements in the MCC score of the GAT model over the sequence baseline for most ligands, while other language model node features have lower relative improvements. Also, the GAT model based on the less complex Prot-BERT embeddings shows higher relative improvement than the more complex Prot-T5 embeddings. Furthermore, more complex node features significantly improve the performance of the sequence baseline. These observations show that the effect of the structure information processed by the GNN is more pronounced for simple node features. This effect decreases with the complexity of the language models. The results for AAIndex then show that the information propagation is a valid way to introduce the structure information. The LM-based GNNs' results suggest that a significant part of the structure information necessary for predicting protein-ligand binding sites is already encoded in the language model embeddings. As the language models were trained on massive amounts of protein sequences, some structural features may already be captured in the embeddings and can thus by used for binding site predictions. This probably explains why the structural information coming from the GNN contributes less strongly to the predictive performance with more complex language models.

4 Conclusion

In this work, we merged sequence-based and structure-based paradigms for protein-ligand binding sites prediction by designing a Graph Neural Network model augmented by protein language model embeddings. Although the model's performance depends on the cutoff distance used to construct the protein graph, the introduction of the graph attention mechanism plays a significant role in improving the predictive performance for densely connected graphs. We have also shown that ensembling GAT models trained on different cutoff distances improves the state-of-the-art performance of the sequence-based pLMs. Finally, we have shown that the contribution of the structural information processed by the GNN architecture to the performance of the model was more significant for simple node features and became less significant as more complex language models were used. Overall, our research demonstrated the potential utility of integrating sequence-based and structure-based approaches, specifically through the use of a Graph Neural Network model augmented by protein language model embeddings, to enhance protein-ligand binding site prediction.

5 Data availability

The source code and data underlying this article are available in the following GitHub repository `https://github.com/hamzagamouh/pt-lm-gnn`.

6 Competing interests

No competing interest is declared.

7 Author contributions statement

D.H. and H.G. conceived the project, D.H. supervised the project, H.G. implemented the method and conducted the experiments, H.G., D.H., and M.N. analysed the results. H. G. wrote most of the manuscript, H.G., D.H., and M.N. reviewed the manuscript.

8 Acknowledgments

This work was supported by the Czech Science Foundation grant 23-07349S. Computational resources were provided by the e-INFRA CZ project (ID:90254), supported by the Ministry of Education, Youth and Sports of the Czech Republic.

References

1. Aaindex database python library. `https://github.com/amckenna41/aaindex`. accessed on 21.06.2023.
2. Bio-embeddings python library. `https://docs.bioembeddings.com/v0.2.3/`. accessed on 21.06.2023.
3. Biopython python library. `https://biopython.org/`. accessed on 21.06.2023.
4. Deep graph library (dgl). `https://www.dgl.ai/`. accessed on 21.06.2023.
5. Dgl-lifesci: Bringing graph neural networks to chemistry and biology. `https://lifesci.dgl.ai/`. accessed on 21.06.2023.

6. Pytorch python library. `https://pytorch.org/`. accessed on 21.06.2023.
7. Rishal Aggarwal, Akash Gupta, Vineeth Chelur, CV Jawahar, and U Deva Priyakumar. Deeppocket: ligand binding site detection and segmentation using 3d convolutional neural networks. *Journal of Chemical Information and Modeling*, 62(21):5069–5079, 2021.
8. Babak Alipanahi, Andrew Delong, Matthew T Weirauch, and Brendan J Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831–838, 2015.
9. Haim Ashkenazy, Elana Erez, Eric Martz, Tal Pupko, and Nir Ben-Tal. Consurf 2010: calculating evolutionary conservation in sequence and structure of proteins and nucleic acids. *Nucleic acids research*, 38(suppl_2):W529–W533, 2010.
10. Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.
11. Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
12. Michal Brylinski and Jeffrey Skolnick. A threading-based method (findsite) for ligand-binding site prediction and functional annotation. *Proceedings of the National Academy of sciences*, 105(1):129–134, 2008.
13. Junyi Chai, Hao Zeng, Anming Li, and Eric WT Ngai. Deep learning in computer vision: A critical review of emerging techniques and application scenarios. *Machine Learning with Applications*, 6:100134, 2021.
14. Jagat S Chauhan, Nitish K Mishra, and Gajendra PS Raghava. Identification of atp binding residues of a protein from its primary sequence. *BMC bioinformatics*, 10:1–9, 2009.
15. Ke Chen, Marcin J Mizianty, and Lukasz Kurgan. Prediction and analysis of nucleotide-binding residues using sequence and sequence-derived structural descriptors. *Bioinformatics*, 28(3):331–341, 2012.
16. Peng Chen, ShanShan Hu, Jun Zhang, Xin Gao, Jinyan Li, Junfeng Xia, and Bing Wang. A sequence-based dynamic ensemble learning system for protein ligand-binding site prediction. *IEEE/ACM transactions on computational biology and bioinformatics*, 13(5):901–912, 2015.
17. Peng Chen, Jianhua Z Huang, and Xin Gao. Ligandrfs: random forest ensemble to identify ligand-binding residues from sequence information alone. In *BMC bioinformatics*, volume 15, pages 1–12. BioMed Central, 2014.
18. Davide Chicco and Giuseppe Jurman. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21(1):6, Dec 2020.
19. Yifeng Cui, Qiwen Dong, Daocheng Hong, and Xikun Wang. Predicting protein-ligand binding residues with deep convolutional neural networks. *BMC bioinformatics*, 20(1):1–12, 2019.
20. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
21. Yijie Ding, Jijun Tang, and Fei Guo. Identification of protein–ligand binding sites by sequence information and ensemble classifier. *Journal of Chemical Information and Modeling*, 57(12):3149–3161, Dec 2017.

22. Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7112–7127, 2021.

23. Sergei A Evteev, Alexey V Ereshchenko, and Yan A Ivanenkov. Siteradar: Utilizing graph machine learning for precise mapping of protein–ligand-binding sites. *Journal of Chemical Information and Modeling*, 63(4):1124–1132, 2023.

24. Leonardo G Ferreira, Ricardo N Dos Santos, Glaucius Oliva, and Adriano D Andricopulo. Molecular docking and structure-based drug design strategies. *Molecules*, 20(7):13384–13421, 2015.

25. Noelia Ferruz and Birte Höcker. Controllable protein design with language models. *Nature Machine Intelligence*, 4(6):521–532, 2022.

26. Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. Protein interface prediction using graph convolutional networks. *Advances in neural information processing systems*, 30, 2017.

27. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

28. Manfred Hendlich, Friedrich Rippmann, and Gerhard Barnickel. Ligsite: automatic and efficient detection of potential small molecule-binding sites in proteins. *Journal of Molecular Graphics and Modelling*, 15(6):359–363, 1997.

29. Magnus Haraldson Høie, Erik Nicolas Kiehl, Bent Petersen, Morten Nielsen, Ole Winther, Henrik Nielsen, Jeppe Hallgren, and Paolo Marcatili. Netsurfp-3.0: accurate and fast prediction of protein structural features by protein language models and deep learning. *Nucleic acids research*, 50(W1):W510–W515, 2022.

30. David Hoksza and Hamza Gamouh. Exploration of protein sequence embeddings for protein-ligand binding site detection. In *2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 3356–3361. IEEE, 2022.

31. Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

32. Kanchan Jha, Sourav Karmakar, and Sriparna Saha. Graph-bert and language model-based framework for protein–protein interaction identification. *Scientific Reports*, 13(1):5663, 2023.

33. José Jiménez, Stefan Doerr, Gerard Martínez-Rosell, Alexander S Rose, and Gianni De Fabritiis. Deepsite: protein-binding site predictor using 3d-convolutional neural networks. *Bioinformatics*, 33(19):3036–3042, 2017.

34. John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

35. Jeevan Kandel, Hilal Tayara, and Kil To Chong. Puresnet: prediction of protein-ligand binding sites using deep residual neural network. *Journal of cheminformatics*, 13(1):1–14, 2021.

36. Shuichi Kawashima and Minoru Kanehisa. Aaindex: amino acid index database. *Nucleic acids research*, 28(1):374–374, 2000.

37. Diksha Khurana, Aditya Koli, Kiran Khatter, and Sukhdev Singh. Natural language processing: State of the art, current trends and challenges. *Multimedia tools and applications*, 82(3):3713–3744, 2023.
38. Pora Kim, Junfei Zhao, Pinyi Lu, and Zhongming Zhao. mutlbsgenedb: mutated ligand binding site gene database. *Nucleic acids research*, 45(D1):D256–D263, 2017.
39. Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
40. Janez Konc and Dušanka Janežič. Protein binding sites for drug design. *Biophysical Reviews*, pages 1–9, 2022.
41. Radoslav Krivák and David Hoksza. P2rank: machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure. *Journal of cheminformatics*, 10:1–12, 2018.
42. Roman A Laskowski. Surfnet: a program for visualizing molecular surfaces, cavities, and intermolecular interactions. *Journal of molecular graphics*, 13(5):323–330, 1995.
43. Alasdair TR Laurie and Richard M Jackson. Q-sitefinder: an energy-based method for the prediction of protein–ligand binding sites. *Bioinformatics*, 21(9):1908–1916, 2005.
44. Vincent Le Guilloux, Peter Schmidtke, and Pierre Tuffery. Fpocket: an open source platform for ligand pocket detection. *BMC bioinformatics*, 10(1):1–11, 2009.
45. Yu Li, Chao Huang, Lizhong Ding, Zhongxiao Li, Yijie Pan, and Xin Gao. Deep learning in bioinformatics: Introduction, application, and perspective in the big data era. *Methods*, 166:4–21, 2019.
46. Yingjie Lin, Seungyeul Yoo, and Roberto Sanchez. Sitecomp: a server for ligand binding site analysis in protein structures. *Bioinformatics*, 28(8):1172–1173, 2012.
47. Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Sal Candido, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *BioRxiv*, 2022.
48. Yang Liu, Maximilian Grimm, Wen-tao Dai, Mu-chun Hou, Zhi-Xiong Xiao, and Yang Cao. Cb-dock: A web server for cavity detection-guided protein–ligand blind docking. *Acta Pharmacologica Sinica*, 41(1):138–144, 2020.
49. Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
50. Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heinz, and Dan Roth. Recent advances in natural language processing via large pre-trained language models: A survey. *arXiv preprint arXiv:2111.01243*, 2021.
51. Stelios K Mylonas, Apostolos Axenopoulos, and Petros Daras. Deepsurf: a surface-based deep learning approach for the prediction of ligand binding sites on proteins. *Bioinformatics*, 37(12):1681–1690, 2021.
52. Chi-Ho Ngan, David R Hall, Brandon Zerbe, Laurie E Grove, Dima Kozakov, and Sandor Vajda. Ftsite: high accuracy detection of ligand binding sites on unbound protein structures. *Bioinformatics*, 28(2):286–287, 2012.
53. Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.
54. Limeng Pu, Rajiv Gandhi Govindaraj, Jeffrey Mitchell Lemoine, Hsiao-Chun Wu, and Michal Brylinski. Deepdrug3d: classification of ligand-binding pockets in proteins with a convolutional neural network. *PLoS computational biology*, 15(2):e1006718, 2019.

55. Roshan Rao, Joshua Meier, Tom Sercu, Sergey Ovchinnikov, and Alexander Rives. Transformer protein language models are unsupervised structure learners. *Biorxiv*, pages 2020–12, 2020.

56. Daniel B Roche, Stuart J Tetchner, and Liam J McGuffin. Funfold: an improved automated method for the prediction of ligand binding residues using 3d models of proteins. *BMC bioinformatics*, 12(1):1–20, 2011.

57. Daniel Barry Roche, Danielle Allison Brackenridge, and Liam James McGuffin. Proteins and their interacting partners: An introduction to protein–ligand binding site prediction methods. *International journal of molecular sciences*, 16(12):29829–29842, 2015.

58. T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks. Mar 2023. arXiv:2303.10993 [cs].

59. Angela Serra, Paola Galdi, and Roberto Tagliaferri. Machine learning for bioinformatics and neuroimaging. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(5):e1248, 2018.

60. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

61. Martin Steinegger, Milot Mirdita, and Johannes Söding. Protein-level assembly increases protein sequence recovery from metagenomic samples manyfold. *Nature methods*, 16(7):603–606, 2019.

62. Baris E Suzek, Yuqi Wang, Hongzhan Huang, Peter B McGarvey, Cathy H Wu, and UniProt Consortium. Uniref clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, 2015.

63. Basant K Tiwary and Basant K Tiwary. Biological databases. *Bioinformatics and Computational Biology: A Primer for Biologists*, pages 11–31, 2022.

64. Serbulent Unsal, Heval Atas, Muammer Albayrak, Kemal Turhan, Aybar C Acar, and Tunca Doğan. Learning functional properties of proteins with language models. *Nature Machine Intelligence*, 4(3):227–245, 2022.

65. Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, et al. Alphafold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research*, 50(D1):D439–D444, 2022.

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

67. Petar Veličković. Everything is connected: Graph neural networks. *Current Opinion in Structural Biology*, 79:102538, 2023.

68. Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

69. Wei Wang, Bin Sun, MengXue Yu, ShiYu Wu, Dong Liu, Hongjun Zhang, and Yun Zhou. Graphplbr: Protein-ligand binding residue prediction with deep graph convolution network. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2023.

70. Yanbin Wang, Zhu-Hong You, Shan Yang, Xiao Li, Tong-Hai Jiang, and Xi Zhou. A high efficient biological language model for predicting protein–protein interactions. *Cells*, 8(2):122, 2019.

71. Mark N Wass, Lawrence A Kelley, and Michael JE Sternberg. 3dligandsite: predicting ligand-binding sites using similar structures. *Nucleic acids research*, 38(suppl_2):W469–W473, 2010.

72. Jianyi Yang, Ambrish Roy, and Yang Zhang. Biolip: a semi-manually curated database for biologically relevant ligand–protein interactions. *Nucleic acids research*, 41(D1):D1096–D1103, 2012.

73. Jianyi Yang, Ambrish Roy, and Yang Zhang. Protein–ligand binding site recognition using complementary binding-specific substructure comparison and sequence profile alignment. *Bioinformatics*, 29(20):2588–2595, 2013.

74. Dong-Jun Yu, Jun Hu, Jing Yang, Hong-Bin Shen, Jinhui Tang, and Jing-Yu Yang. Designing template-free predictor for targeting protein-ligand binding sites with classifier ensemble and spatial clustering. *IEEE/ACM transactions on computational biology and bioinformatics*, 10(4):994–1008, 2013.

75. Xiao-Meng Zhang, Li Liang, Lin Liu, and Ming-Jing Tang. Graph neural networks and their current applications in bioinformatics. *Frontiers in genetics*, 12:690049, 2021.

76. Yang Zhang, Wenbing Huang, Zhewei Wei, Ye Yuan, and Zhaohan Ding. Equipocket: an e (3)-equivariant geometric graph neural network for ligand binding site prediction. *arXiv preprint arXiv:2302.12177*, 2023.

77. Jingtian Zhao, Yang Cao, and Le Zhang. Exploring the computational methods for protein-ligand binding site prediction. *Computational and structural biotechnology journal*, 18:417–426, 2020.

78. Ziqi Zhao, Yonghong Xu, and Yong Zhao. Sxgbsite: Prediction of protein–ligand binding sites using sequence information and extreme gradient boosting. *Genes*, 10(12):965, Nov 2019.

79. Zaixiang Zheng, Yifan Deng, Dongyu Xue, Yi Zhou, Fei Ye, and Quanquan Gu. Structure-informed language models are protein designers. *bioRxiv*, pages 2023–02, 2023.

9 Supplementary material

Ligand	Shallow GCN							Deep GCN						
	Split 0	Split 1	Split 2	Split 3	Split 4	Mean	STD	Split 0	Split 1	Split 2	Split 3	Split 4	Mean	STD
ADP	0.564	0.587	0.566	0.571	0.573	0.572	0.009	0.530	0.558	0.553	0.540	0.557	0.548	0.012
AMP	0.412	0.441	0.431	0.446	0.453	0.437	0.016	0.416	0.448	0.460	0.448	0.426	0.440	0.018
ATP	0.537	0.557	0.552	0.532	0.554	0.547	0.011	0.521	0.529	0.559	0.551	0.544	0.541	0.016
CA	0.382	0.399	0.399	0.405	0.394	0.396	0.009	0.334	0.349	0.341	0.334	0.327	0.337	0.008
DNA	0.476	0.449	0.450	0.454	0.473	0.460	0.013	0.453	0.456	0.452	0.481	0.441	0.457	0.015
FE	0.645	0.622	0.565	0.619	0.615	0.613	0.030	0.649	0.654	0.650	0.679	0.619	0.650	0.021
GDP	0.668	0.691	0.661	0.655	0.677	0.671	0.014	0.535	0.541	0.663	0.595	0.624	0.592	0.055
GTP	0.514	0.452	0.573	0.479	0.540	0.511	0.048	0.545	0.453	0.528	0.472	0.511	0.502	0.038
HEME	0.672	0.706	0.729	0.741	0.706	0.711	0.026	0.662	0.699	0.709	0.727	0.724	0.704	0.026
MG	0.344	0.356	0.350	0.346	0.339	0.347	0.006	0.282	0.278	0.280	0.280	0.322	0.288	0.019
MN	0.606	0.609	0.607	0.597	0.617	0.607	0.007	0.587	0.558	0.571	0.569	0.572	0.572	0.010
ZN	0.681	0.664	0.677	0.675	0.682	0.676	0.007	0.627	0.624	0.629	0.629	0.644	0.631	0.008

Note : The displayed scores are test MCC scores.

Table 5. Different random splits for the GCN model with ProtT5 embeddings and cutoff distance of 6 Å

		GCN				GAT			
Ligand	Cutoff	AUC	Precision	Recall	MCC	AUC	Precision	Recall	MCC
ADP	4	0.879	0.663	0.51	0.569	0.93	0.674	0.504	0.571
	6	0.874	0.658	0.504	0.564	0.932	0.704	0.493	0.578
	8	0.881	0.656	0.536	0.581	0.945	0.674	0.55	0.597
	10	0.872	0.661	0.491	0.557	0.949	0.641	0.551	0.582
	Ensemble	0.907	0.726	0.487	0.584	0.952	0.713	0.496	0.583
AMP	4	0.818	0.49	0.449	0.45	0.899	0.398	0.561	0.449
	6	0.789	0.443	0.423	0.412	0.892	0.465	0.503	0.463
	8	0.766	0.448	0.441	0.424	0.892	0.67	0.378	0.489
	10	0.803	0.458	0.421	0.419	0.892	0.618	0.388	0.475
	Ensemble	0.838	0.54	0.395	0.445	0.903	0.632	0.39	0.482
ATP	4	0.884	0.578	0.547	0.546	0.936	0.567	0.598	0.566
	6	0.876	0.543	0.567	0.537	0.931	0.651	0.533	0.575
	8	0.879	0.554	0.558	0.538	0.936	0.62	0.556	0.572
	10	0.89	0.579	0.567	0.557	0.946	0.66	0.547	0.587
	Ensemble	0.914	0.651	0.522	0.569	0.946	0.677	0.526	0.583
CA	4	0.789	0.526	0.308	0.396	0.828	0.473	0.322	0.383
	6	0.784	0.531	0.284	0.382	0.826	0.522	0.329	0.408
	8	0.77	0.551	0.303	0.403	0.822	0.533	0.322	0.408
	10	0.783	0.595	0.304	0.42	0.834	0.524	0.332	0.411
	Ensemble	0.804	0.647	0.28	0.421	0.842	0.613	0.303	0.426
DNA	4	0.873	0.48	0.53	0.473	0.912	0.442	0.552	0.46
	6	0.891	0.476	0.541	0.476	0.922	0.459	0.579	0.483
	8	0.883	0.411	0.625	0.47	0.932	0.438	0.674	0.51
	10	0.885	0.43	0.568	0.459	0.929	0.474	0.569	0.488
	Ensemble	0.907	0.519	0.518	0.49	0.934	0.5	0.559	0.499
FE	4	0.934	0.463	0.842	0.618	0.976	0.602	0.833	0.704
	6	0.935	0.488	0.867	0.645	0.982	0.572	0.792	0.668
	8	0.938	0.467	0.825	0.614	0.986	0.594	0.817	0.692
	10	0.95	0.493	0.858	0.645	0.987	0.609	0.858	0.719
	Ensemble	0.948	0.513	0.825	0.645	0.983	0.613	0.817	0.703
GDP	4	0.878	0.756	0.608	0.665	0.954	0.851	0.588	0.696
	6	0.887	0.782	0.593	0.668	0.96	0.801	0.624	0.695
	8	0.886	0.879	0.634	0.737	0.963	0.937	0.608	0.746
	10	0.891	0.796	0.624	0.693	0.969	0.735	0.701	0.705
	Ensemble	0.924	0.896	0.577	0.71	0.971	0.922	0.613	0.744
GTP	4	0.851	0.469	0.674	0.537	0.917	0.761	0.607	0.666
	6	0.861	0.47	0.618	0.514	0.902	0.753	0.618	0.669
	8	0.866	0.553	0.64	0.575	0.932	0.731	0.64	0.67
	10	0.873	0.544	0.629	0.564	0.909	0.508	0.697	0.573
	Ensemble	0.885	0.58	0.573	0.556	0.926	0.809	0.618	0.695
HEME	4	0.927	0.716	0.7	0.689	0.965	0.715	0.676	0.675
	6	0.929	0.718	0.667	0.672	0.962	0.773	0.621	0.674
	8	0.948	0.755	0.75	0.736	0.976	0.799	0.719	0.743
	10	0.935	0.716	0.674	0.675	0.97	0.724	0.679	0.682
	Ensemble	0.965	0.775	0.648	0.691	0.976	0.79	0.624	0.685
MG	4	0.735	0.459	0.264	0.343	0.754	0.44	0.249	0.325
	6	0.731	0.443	0.276	0.344	0.77	0.471	0.264	0.347
	8	0.737	0.438	0.292	0.351	0.782	0.473	0.289	0.364
	10	0.723	0.481	0.281	0.362	0.771	0.463	0.272	0.349
	Ensemble	0.755	0.526	0.261	0.365	0.783	0.537	0.254	0.364
MN	4	0.891	0.6	0.646	0.617	0.919	0.625	0.591	0.602
	6	0.905	0.565	0.662	0.606	0.92	0.684	0.612	0.642
	8	0.871	0.596	0.603	0.594	0.92	0.608	0.616	0.607
	10	0.874	0.55	0.646	0.59	0.931	0.652	0.641	0.642
	Ensemble	0.913	0.64	0.637	0.634	0.927	0.679	0.608	0.638
ZN	4	0.95	0.64	0.691	0.66	0.958	0.667	0.683	0.67
	6	0.951	0.711	0.661	0.681	0.959	0.681	0.672	0.672
	8	0.944	0.685	0.671	0.673	0.962	0.712	0.668	0.685
	10	0.948	0.729	0.667	0.693	0.964	0.713	0.677	0.69
	Ensemble	0.956	0.755	0.655	0.699	0.966	0.746	0.663	0.699

Table 6. Comparison of GAT and GCN for ProtT5 embeddings and for different cutoff distances

		AAIndex				ProtBERT				ProtT5			
Ligand	Cutoff	AUC	Precision	Recall	MCC	AUC	Precision	Recall	MCC	AUC	Precision	Recall	MCC
ADP	4	0.664	0.055	0.557	0.089	0.841	0.755	0.324	0.484	0.930	0.674	0.504	0.571
	6	0.675	0.075	0.436	0.115	0.868	0.728	0.340	0.487	0.932	0.704	0.493	0.578
	8	0.735	0.077	0.583	0.142	0.885	0.659	0.405	0.504	0.945	0.674	0.550	0.597
	10	0.750	0.084	0.573	0.154	0.891	0.705	0.359	0.492	0.949	0.641	0.551	0.582
	Ensemble	0.756	0.127	0.430	0.188	0.902	0.807	0.324	0.502	0.952	0.713	0.496	0.583
AMP	4	0.642	0.050	0.648	0.067	0.852	0.233	0.429	0.281	0.899	0.398	0.561	0.449
	6	0.668	0.060	0.571	0.092	0.859	0.365	0.235	0.272	0.892	0.465	0.503	0.463
	8	0.684	0.060	0.643	0.101	0.865	0.321	0.388	0.326	0.892	0.670	0.378	0.489
	10	0.703	0.058	0.737	0.108	0.879	0.313	0.370	0.313	0.892	0.618	0.388	0.475
	Ensemble	0.718	0.068	0.569	0.111	0.880	0.359	0.278	0.293	0.903	0.632	0.390	0.482
ATP	4	0.635	0.054	0.567	0.071	0.871	0.541	0.366	0.428	0.936	0.567	0.598	0.566
	6	0.681	0.067	0.573	0.106	0.880	0.441	0.490	0.443	0.931	0.651	0.533	0.575
	8	0.690	0.073	0.470	0.106	0.884	0.449	0.484	0.445	0.936	0.620	0.556	0.572
	10	0.689	0.075	0.516	0.116	0.894	0.445	0.539	0.468	0.946	0.660	0.547	0.587
	Ensemble	0.702	0.092	0.454	0.136	0.904	0.552	0.440	0.476	0.946	0.677	0.526	0.583
CA	4	0.749	0.038	0.615	0.110	0.796	0.481	0.248	0.339	0.828	0.473	0.322	0.383
	6	0.753	0.038	0.641	0.112	0.804	0.543	0.233	0.350	0.826	0.522	0.329	0.408
	8	0.766	0.045	0.575	0.122	0.805	0.598	0.246	0.378	0.822	0.533	0.322	0.408
	10	0.785	0.048	0.611	0.134	0.819	0.547	0.237	0.354	0.834	0.524	0.332	0.411
	Ensemble	0.791	0.054	0.558	0.139	0.817	0.661	0.214	0.372	0.842	0.613	0.303	0.426
DNA	4	0.685	0.094	0.693	0.138	0.860	0.354	0.461	0.363	0.912	0.442	0.552	0.460
	6	0.704	0.103	0.629	0.145	0.877	0.352	0.503	0.379	0.922	0.459	0.579	0.483
	8	0.731	0.118	0.597	0.168	0.880	0.323	0.562	0.381	0.932	0.438	0.674	0.510
	10	0.741	0.124	0.622	0.184	0.878	0.355	0.499	0.380	0.929	0.474	0.569	0.488
	Ensemble	0.734	0.124	0.584	0.175	0.891	0.405	0.455	0.393	0.934	0.500	0.559	0.499
FE	4	0.893	0.061	0.808	0.194	0.973	0.557	0.692	0.615	0.976	0.602	0.833	0.704
	6	0.895	0.089	0.675	0.221	0.984	0.535	0.767	0.635	0.982	0.572	0.792	0.668
	8	0.897	0.090	0.775	0.241	0.983	0.535	0.767	0.635	0.986	0.594	0.817	0.692
	10	0.912	0.084	0.808	0.237	0.985	0.458	0.825	0.609	0.987	0.609	0.858	0.719
	Ensemble	0.917	0.121	0.733	0.279	0.985	0.544	0.725	0.622	0.983	0.613	0.817	0.703
GDP	4	0.698	0.089	0.531	0.130	0.903	0.624	0.546	0.566	0.954	0.851	0.588	0.696
	6	0.742	0.115	0.474	0.163	0.924	0.677	0.552	0.595	0.960	0.801	0.624	0.695
	8	0.776	0.156	0.479	0.215	0.931	0.675	0.526	0.580	0.963	0.937	0.608	0.746
	10	0.786	0.145	0.593	0.230	0.930	0.603	0.557	0.561	0.969	0.735	0.701	0.705
	Ensemble	0.776	0.201	0.438	0.249	0.945	0.768	0.495	0.603	0.971	0.922	0.613	0.744
GTP	4	0.685	0.088	0.551	0.129	0.907	0.694	0.483	0.563	0.917	0.761	0.607	0.666
	6	0.704	0.122	0.517	0.180	0.884	0.658	0.562	0.591	0.902	0.753	0.618	0.669
	8	0.759	0.133	0.506	0.192	0.880	0.643	0.506	0.552	0.932	0.731	0.640	0.670
	10	0.711	0.097	0.562	0.149	0.905	0.613	0.551	0.562	0.909	0.508	0.697	0.573
	Ensemble	0.749	0.184	0.449	0.235	0.904	0.789	0.506	0.618	0.926	0.809	0.618	0.695
HEME	4	0.690	0.105	0.612	0.131	0.904	0.577	0.472	0.493	0.965	0.715	0.676	0.675
	6	0.724	0.117	0.617	0.156	0.912	0.593	0.495	0.514	0.962	0.773	0.621	0.674
	8	0.760	0.142	0.681	0.214	0.926	0.617	0.566	0.564	0.976	0.799	0.719	0.743
	10	0.750	0.136	0.616	0.190	0.925	0.552	0.547	0.519	0.970	0.724	0.679	0.682
	Ensemble	0.763	0.153	0.572	0.206	0.936	0.699	0.476	0.554	0.976	0.790	0.624	0.685
MG	4	0.688	0.024	0.546	0.068	0.747	0.447	0.244	0.325	0.754	0.440	0.249	0.325
	6	0.689	0.024	0.482	0.065	0.745	0.480	0.227	0.325	0.770	0.471	0.264	0.347
	8	0.707	0.032	0.437	0.083	0.751	0.412	0.264	0.324	0.782	0.473	0.289	0.364
	10	0.701	0.029	0.482	0.078	0.756	0.477	0.232	0.328	0.771	0.463	0.272	0.349
	Ensemble	0.710	0.040	0.417	0.098	0.769	0.519	0.219	0.333	0.783	0.537	0.254	0.364
MN	4	0.840	0.051	0.776	0.165	0.880	0.511	0.506	0.502	0.919	0.625	0.591	0.602
	6	0.843	0.076	0.624	0.191	0.890	0.552	0.473	0.505	0.920	0.684	0.612	0.642
	8	0.857	0.079	0.667	0.204	0.901	0.544	0.498	0.514	0.920	0.608	0.616	0.607
	10	0.863	0.085	0.646	0.210	0.909	0.543	0.527	0.529	0.931	0.652	0.641	0.642
	Ensemble	0.869	0.106	0.662	0.243	0.909	0.591	0.481	0.527	0.927	0.679	0.608	0.638
ZN	4	0.934	0.096	0.895	0.268	0.959	0.640	0.578	0.602	0.958	0.667	0.683	0.670
	6	0.934	0.131	0.798	0.303	0.954	0.648	0.579	0.607	0.959	0.681	0.672	0.672
	8	0.941	0.152	0.810	0.331	0.960	0.640	0.609	0.618	0.962	0.712	0.668	0.685
	10	0.939	0.146	0.852	0.334	0.959	0.636	0.612	0.618	0.964	0.713	0.677	0.690
	Ensemble	0.942	0.165	0.816	0.349	0.962	0.687	0.570	0.621	0.966	0.746	0.663	0.699

Note : Here, node features are compared for the GAT model.

Table 7. Comparison of different node features with different cutoff distances