DrugHash: Hashing Based Contrastive Learning for Virtual Screening

Jin Han[1*], **Yun Hong**[2*], **Wu-Jun Li**[1†]

[1] National Key Laboratory for Novel Software Technology, School of Computer Science, Nanjing University
[2] Kuang Yaming Honors School, Nanjing University
{hanjin, hongy}@smail.nju.edu.cn, liwujun@nju.edu.cn

Abstract

Virtual screening (VS) is a critical step in computer-aided drug discovery, aiming to identify molecules that bind to a specific target protein. Traditional VS methods, such as docking, are often too time-consuming to efficiently screen large-scale molecular databases. Recent advances in deep learning have demonstrated that learning vector representations for both proteins and molecules using contrastive learning can outperform traditional docking methods. However, considering that the target databases often contain billions of molecules, real-valued vector representations adopted by existing methods can still incur large memory and time cost in VS. To address this problem, we propose DrugHash, a hashing-based contrastive learning method for VS. DrugHash formulates VS as a retrieval task that leverages binary hash codes for efficient retrieval. In particular, DrugHash designs a simple yet effective hashing strategy to enable end-to-end learning of binary hash codes for both proteins and molecules, which can dramatically reduce the memory and time cost with higher accuracy compared with existing methods. Experimental results show that DrugHash can outperform existing methods to achieve state-of-the-art accuracy, with at least a 32× reduction in memory cost and a 4.6× improvement in speed.

Introduction

Due to the high failure rate of drug candidates, drug discovery requires a long development cycle and substantial cost (Gorgulla et al. 2020). Virtual screening (VS) is a critical step in computer-aided drug discovery, aiming to identify molecules that bind to a specific target protein. High-quality VS can streamline the drug discovery process by identifying promising lead compounds, thereby reducing both time and resource cost. One of the key factors influencing the effectiveness of VS is the size of the molecular database. A larger database generally includes more potential binding molecules, increasing the likelihood of identifying effective drug candidates. However, as shown in (Lyu et al. 2019), a poorly performing VS method will increase the false positive rate when the database is enlarged. This highlights two requirements for effective VS: a large-scale molecular database and an effective VS algorithm.

Numerous large-scale molecular databases have been developed for drug discovery. For instance, ZINC (Irwin and Shoichet 2005), a widely used commercial database, contains over 1.4 billion compounds in its latest release called ZINC20 (Irwin et al. 2020). The Enamine REAL database (Shivanyuk et al. 2007) contains over 6.75 billion molecules. Despite the availability of these extensive databases, existing VS methods often struggle to handle their scale. Traditional VS methods, such as molecular docking methods, are particularly time-consuming when screening large molecular databases. For instance, AutoDock Vina (Kitchen et al. 2004) requires around 70 seconds to dock a single molecule with a single-core CPU when the exhaustiveness is set to 8. Hence, docking millions of molecules, the scale of which is comparable with or even smaller than those in many real applications, would take Vina over two years. While increasing the number of CPUs can reduce docking time, it comes with substantial resource consumption.

In contrast, the learning-based VS methods (Zheng, Fan, and Mu 2019; Jones et al. 2021; Zhang et al. 2023; Wu et al. 2022; Yazdani-Jahromi et al. 2022) are relatively more resource-efficient. However, most of the existing methods focus on predicting the binding affinity or interactions between the proteins and molecules, As a result, they often fail to outperform traditional docking methods in VS benchmarks. Recently, DrugCLIP (Gao et al. 2023) proposes to formulate VS as a retrieval task rather than learning the binding affinity or protein-molecule interaction. Through contrastive learning, DrugCLIP learns real-valued vector representations for both proteins and molecules, enabling it to surpass traditional docking methods. However, the use of real-valued vector representations for molecular databases, particularly those containing billions of molecules, presents a significant challenge on memory consumption. For example, DrugCLIP encodes molecules into 128-dimensional real-valued vectors, and hence the real-valued vector representations for the Enamine REAL database would have a memory cost of nearly 3TB, which is extremely large for local computer memory. Additionally, loading pre-encoded vectors from storage, computing similarity scores between billions of vectors, and ranking the results can be time-

[*] These authors contributed equally.
[†] Corresponding author

consuming.

In this paper, we propose a hashing-based contrastive learning method, called DrugHash, for VS. DrugHash also treats VS as a retrieval task, but it uses binary hash codes for efficient retrieval. The contributions of DrugHash are outlined as follows:

- To the best of our knowledge, DrugHash is the first hashing method for VS.
- DrugHash designs a simple yet effective hashing strategy to enable end-to-end learning of binary hash codes for both protein and molecule modalities, which can dramatically reduce the memory and time cost.
- DrugHash can also outperform existing methods in terms of accuracy. This seems to be counter-intuitive but is actually reasonable, because binary hash codes can act as a constraint (regularization) for improving generalization ability.
- Experimental results show that DrugHash can outperform existing methods to achieve state-of-the-art accuracy, with at least a $32\times$ reduction in memory cost and a $4.6\times$ improvement in speed.

Related Works

Virtual Screening Existing VS methods can be broadly categorized into two main classes: docking-based methods and learning-based methods. Docking-based methods like FlexX (Rarey et al. 1996), Glide (Friesner et al. 2004) and AutoDock Vina (Trott and Olson 2010) involve predicting the molecule conformation and pose given a target protein (Kitchen et al. 2004). Docking relies on complex scoring functions to estimate the binding affinity or strength of the connection between the molecule and the target protein (Raval and Ganatra 2022). Random approaches like Monte Carlo and genetic algorithms are performed to search the vast conformation space to find the optimal docking pose. Hence, docking-based methods are typically very time-consuming.

Most of the learning-based methods are supervised by the binding affinities or interactions between molecules and proteins. Methods such as OnionNet (Zheng, Fan, and Mu 2019), FAST (Jones et al. 2021), Planet (Zhang et al. 2023), and CAPLA (Jin et al. 2023) treat VS as a regression task that predicts the binding affinities of the protein-ligand complex. Another kind of methods such as BridgeDPI (Wu et al. 2022), TransformerCPI (Chen et al. 2020), and AttentionSiteDTI (Yazdani-Jahromi et al. 2022) treat the problem as a binary classification task and predict whether or not a molecule could bind to the target protein. However, due to the scarcity of labeled data, the performance of these supervised methods is limited and falls behind docking-based methods (Brocidiacono et al. 2022). Recently, DrugCLIP (Gao et al. 2023) proposes to formulate VS as a retrieval task. By applying data argumentation and learning real-valued vector representations for proteins and molecules with a contrastive learning objective, DrugCLIP can surpass traditional docking methods on VS benchmarks. However, the real-valued vectors for feature representations will bring large memory and time cost to screen billion-scale molecular databases.

Cross-Modal Hashing Hashing adopts a hash function to map each data point in the database to a binary vector (or called hash code) while preserving the similarity in the original space. In cross-modal hashing (CMH), the modality of the query point is different from the modality of the points in the database, and hence different hash functions will be designed for different modalities (Bronstein et al. 2010; Kumar and Udupa 2011; Lin et al. 2015). Binary hash codes have advantages in low storage cost and fast retrieval time, and hence there are many CMH methods proposed for multimedia data (text, audio, images, video) retrieval (Jiang and Li 2017; Deng et al. 2018; Shen et al. 2020; Hu et al. 2022). To the best of our knowledge, no research has explored hashing and CMH on VS problems including protein-molecule retrieval.

Method

In this section, we present the details of our proposed method called DrugHash, which is a hashing-based contrastive learning method. DrugHash treats VS as a retrieval task. Although the techniques in DrugHash can be used for various VS problems, we focus on protein-ligand complexes (Wang et al. 2005; Yang, Roy, and Zhang 2012; Gaulton et al. 2012) in this paper. Hence, the term protein actually refers to the protein pocket. For brevity, we will continue to use protein in the following content. The term ligand refers to a small molecule that binds to a target protein. The complexes reveal the structural and functional relationships between proteins and molecules. We treat the complexes as data of two modalities: proteins and molecules. DrugHash uses protein queries to retrieve molecules in the database.

The hash function can be manually designed or learned from the training data. We focus on learning hash functions for proteins and molecules. Figure 1 illustrates the architecture of DrugHash[1]. DrugHash contains the protein and molecule encoder and the objective function which includes a contrastive learning objective and a cross-modal hashing objective.

Protein and Molecule Encoder

Various protein and molecule encoders can be adopted in DrugHash. To show the effectiveness of our proposed hashing strategy, we adopt the same encoder as DrugCLIP (Gao et al. 2023), which is a pre-trained SE(3) Transformer proposed in Uni-Mol (Zhou et al. 2023). The input of the encoder are atom representation which is initialized from atom type and pair representation which is initialized by encoding the Euclidean distance between pairs of atoms using a Gaussian kernel (Shuaibi et al. 2021). We denote the pair representation of ij in layer l as $\boldsymbol{q}_{ij}^{l}$. $\boldsymbol{q}_{ij}^{l}$ is first updated by the Query-Key product in attention mechanism (Vaswani et al.

[1]The protein and molecule pictures in Figure 1 and Figure 2 are from Protein Data Bank (Berman et al. 2000).

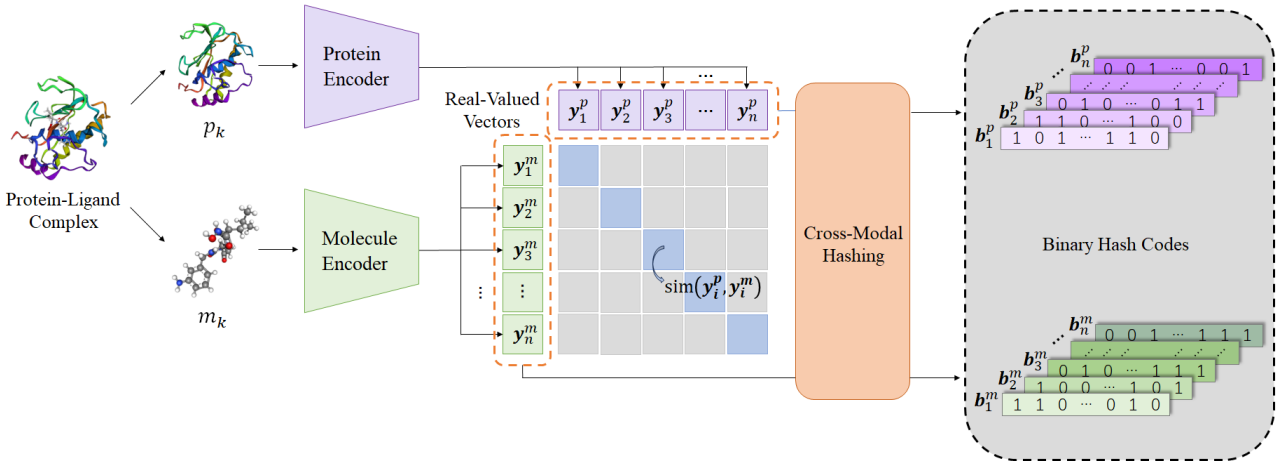

Figure 1: Illustration of the DrugHash architecture.

2017):

$$\boldsymbol{q}_{ij}^{l+1,h} = \boldsymbol{q}_{ij}^{l,h} + \frac{\boldsymbol{Q}_i^{l,h}(\boldsymbol{K}_j^{l,h})^\top}{\sqrt{d}}, h \in [1, H], \tag{1}$$

where $\boldsymbol{Q}_i^{l,h}$ and $\boldsymbol{K}_j^{l,h}$ is the Query and Key of i-th and j-th atom representation in layer l for the h-th attention head, H is number of attention heads, d is the dimension of Key. The pair representation serves as a bias term in self-attention to update the atom representation:

$$\begin{aligned}&\text{Attention}(\boldsymbol{Q}_i^{l,h}, \boldsymbol{K}_j^{l,h}, \boldsymbol{V}_j^{l,h}) \\ &= \text{softmax}(\frac{\boldsymbol{Q}_i^{l,h}(\boldsymbol{K}_j^{l,h})^\top}{\sqrt{d}} + \boldsymbol{q}_{ij}^{l,h})\boldsymbol{V}_j^{l,h},\end{aligned} \tag{2}$$

where $\boldsymbol{V}_j^{l,h}$ is the Value of the j-th atom representation in layer l for the h-th attention head. The above encoder is pre-trained through 3D position recovery and atom-type recovery tasks. The input atom coordinates are randomly corrupted, and the model is trained to predict the correct positions and pairwise distance of the atoms. The atom types of corrupted atoms are also masked by a [CLS] token, and the model is trained to predict the masked atom type. The protein encoder is pre-trained on 3.2 million protein pockets and the molecule encoder is pre-trained on 19 million molecules. The encoder of DrugHash is initialized by the parameter of the trained encoder.

Formally, we denote the above encoding process of protein and molecule as E_p and E_m respectively. Considering a set of n complexes, we denote the set of proteins as $P = \{p_1, p_2, \cdots, p_n\}$ and the set of molecules as $M = \{m_1, m_2, \cdots, m_n\}$. For any index k, p_k and m_k indicate the protein and molecule data originating from the same complex. For a protein-molecule pair (p_k, m_k), their vector representation $(\boldsymbol{y}_k^p, \boldsymbol{y}_k^m)$ can be obtained by:

$$(\boldsymbol{y}_k^p, \boldsymbol{y}_k^m) = (E_p(p_k), E_m(m_k)). \tag{3}$$

Objective Function

The objective function of DrugHash includes a contrastive learning objective and a cross-modal hashing objective. The contrastive learning objective aims to align the representation of proteins and molecules in a shared embedding space. The cross-modal hashing objective aims to enable end-to-end learning of the binary hash codes for both protein and molecule modalities.

Contrastive Learning We aim to maximize the similarity between correct pairs and minimize the similarity between incorrect pairs. In this context, the word "correct" means that the protein and molecule belong to the same complex. The encoded protein set $E_p(P) = \{\boldsymbol{y}_1^p, \boldsymbol{y}_2^p, \cdots, \boldsymbol{y}_n^p\}$ and the encoded molecule set $E_m(M) = \{\boldsymbol{y}_1^m, \boldsymbol{y}_2^m, \cdots, \boldsymbol{y}_n^m\}$. As shown in Figure 1, there are n^2 protein-molecule pairs in total, but only the pairs in the diagonal are supposed to be similar. We use cosine similarity to define the similarity between $\boldsymbol{y}_i^p$ and $\boldsymbol{y}_j^m$:

$$\text{sim}(\boldsymbol{y}_i^p, \boldsymbol{y}_j^m) = (\boldsymbol{y}_i^p)^\top \cdot \boldsymbol{y}_j^m / (\|\boldsymbol{y}_i^p\| \|\boldsymbol{y}_j^m\|). \tag{4}$$

We use infoNCE loss (Oord, Li, and Vinyals 2018) to define our contrastive learning objective, which aims to minimize the negative log-likelihood of similar protein-molecule pairs. From the perspective of protein modality, we aim to distinguish the true ligand that binds to the given protein:

$$\mathcal{L}_k^p = -\frac{1}{n} \log \frac{\exp(\text{sim}(\boldsymbol{y}_k^p, \boldsymbol{y}_k^m)/\tau)}{\sum_i \exp(\text{sim}(\boldsymbol{y}_k^p, \boldsymbol{y}_i^m)/\tau)}, \tag{5}$$

where τ denotes a temperature hyperparameter. From the perspective of molecule modality, we aim to distinguish the true receptor that accepts the given molecule:

$$\mathcal{L}_k^m = -\frac{1}{n} \log \frac{\exp(\text{sim}(\boldsymbol{y}_k^p, \boldsymbol{y}_k^m)/\tau)}{\sum_i \exp(\text{sim}(\boldsymbol{y}_i^p, \boldsymbol{y}_k^m)/\tau)}. \tag{6}$$

The above loss function has been utilized in existing works like CLIP (Radford et al. 2021) and DrugCLIP (Gao et al. 2023). The overall contrastive learning objective is defined as:

$$\mathcal{L}_c = \frac{1}{2} \sum_{k=1}^{n} (\mathcal{L}_k^p + \mathcal{L}_k^m). \tag{7}$$

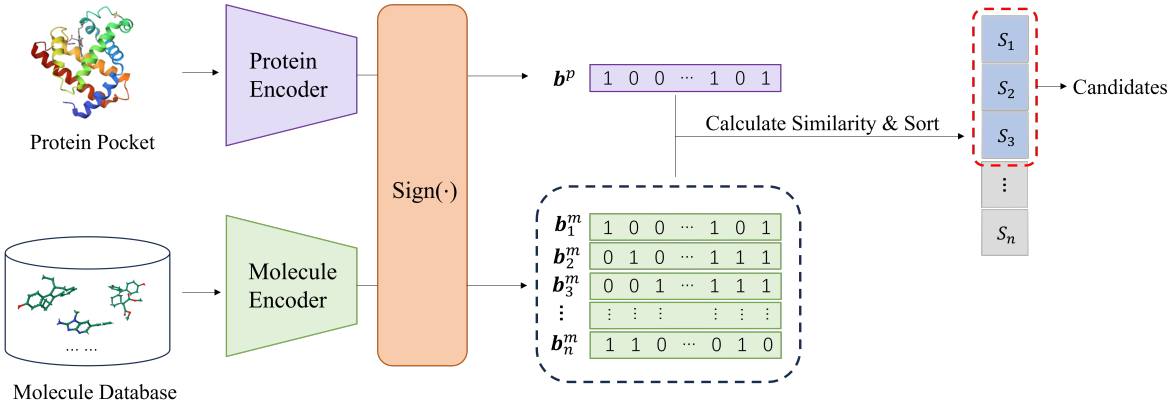

Figure 2: Illustration of the inference stage.

Cross-Modal Hashing Note that the vector representation $\boldsymbol{y}_k^p$ and $\boldsymbol{y}_k^m$ of protein and molecule is still real-valued at this stage. Unlike existing methods, we aim to learn the binary hash codes for both protein and molecule modality. Let $\boldsymbol{b}_k^p \in \{-1, 1\}^d$ and $\boldsymbol{b}_k^m \in \{-1, 1\}^d$ respectively denote the binary hash codes for protein p_k and molecule m_k, where d is the code length and is the same as the embedding dimension of $\boldsymbol{y}_k^p$ and $\boldsymbol{y}_k^m$. We define the loss function for hashing as follows:

$$\mathcal{L}_{\text{hash}} = \frac{1}{nd} \sum_{k=1}^{n} (\|\boldsymbol{y}_k^p - \boldsymbol{b}_k^p\|_2^2 + \|\boldsymbol{y}_k^m - \boldsymbol{b}_k^m\|_2^2). \quad (8)$$

The loss function designed in this way has two purposes. On one hand, it brings $\boldsymbol{y}_k^p$ and $\boldsymbol{y}_k^m$ closer to its binary hash codes $\boldsymbol{b}_k^p$ and $\boldsymbol{b}_k^m$. On the other hand, it serves as a regularization term to alleviate model overfitting.

The overall objective function is formulated as follows:

$$\mathcal{L} = \mathcal{L}_c + \lambda \mathcal{L}_{\text{hash}}, \quad (9)$$

where λ is a hyperparameter for balancing the two loss terms.

Training and Inference

We denote the whole parameter of E_p and E_m as θ. In the training stage, the model parameter θ and binary hash codes $\boldsymbol{b}_k^p$ and $\boldsymbol{b}_k^m$ can be optimized alternately. When θ is fixed, $\boldsymbol{b}_k^p$ and $\boldsymbol{b}_k^m$ can be obtained by

$$\begin{aligned} \boldsymbol{b}_k^p &= \text{sign}(\boldsymbol{y}_k^p), \\ \boldsymbol{b}_k^m &= \text{sign}(\boldsymbol{y}_k^m), \end{aligned} \quad (10)$$

where the function $\text{sign}(\cdot)$ returns the signs of the elements. When the binary hash codes are fixed, θ can be optimized by backpropagation.

In the inference stage, as illustrated in Figure 2, we aim to retrieve molecules that would bind to the target protein from the molecular database. Given a target protein p and molecular database $D_M = \{m_1, m_2, m_3, \cdots\}$, the binary hash code $\boldsymbol{b}^p$ of protein and binary vector database B_{D_m} of molecules can be obtained as follows:

$$\begin{aligned} \boldsymbol{b}^p &= \text{sign}(E_p(p)), \\ B_{D_m} &= \{\boldsymbol{b}_1^m, \boldsymbol{b}_2^m, \cdots\} \\ &= \{\text{sign}(E_m(m_1)), \text{sign}(E_m(m_2)), \cdots\}. \end{aligned} \quad (11)$$

The number -1 in hash codes can be easily changed to 0 to get the final hash code representation. We rank the molecules most likely to bind with the target protein based on Hamming distance, which can be calculated as the number of different bits of binary hash codes.

Experiment

Evaluation Settings

Metric We adopt several metrics to evaluate the accuracy of VS: the area under the receiver operating characteristic curve (AUROC), the Boltzmann-enhanced discrimination of receiver operating characteristic (BEDROC), and the enrichment factor (EF). AUROC is a commonly used metric to evaluate the ranking performance. BEDORC is a metric proposed in (Truchon and Bayly 2007), which is sensitive to "early recognition". EF is formulated as $\text{EF}^{x\%} = \frac{\text{Hits}^{x\%}/\text{N}^{x\%}}{\text{Hits}^{total}/\text{N}^{total}}$, where $\text{Hits}^{x\%}$ and Hits^{total} denote the actives in top $x\%$ and total database, $\text{N}^{x\%}$ and N^{total} denote the number of molecules in top $x\%$ and total database. We consider the $\text{EF}^{x\%}$ of $\text{EF}^{0.5\%}$, $\text{EF}^{1\%}$ and $\text{EF}^{5\%}$. We also calculate the memory and time cost for different VS methods.

Datasets To train DrugHash, we adopt the same training datasets as DrugCLIP, which is the PDBBind dataset (Wang et al. 2005) argumented by HomoAug (Gao et al. 2023). To benchmark the VS performance of different methods, we adopt two evaluation datasets, which are DUD-E (Mysinger et al. 2012) and LIT-PCBA (Tran-Nguyen, Jacquemard, and Rognan 2020). DUD-E contains 102 protein targets, in which each target is associated with a set of ligands and decoys. The total number of ligands is 22,886 and each is with 50 property-matched decoys. LIT-PCBA contains 15 target proteins, and the number of active compounds is 7,844 while

	AUROC	BEDROC	$EF^{0.5\%}$	$EF^{1\%}$	$EF^{5\%}$
Glide-SP	76.70	40.70	19.39	16.18	7.23
Vina	71.60	-	9.13	7.32	4.44
NN-score	68.30	12.20	4.16	4.02	3.12
RF-Score	65.21	12.41	4.90	4.52	2.98
Pafnucy	63.11	16.50	4.24	3.86	3.76
Planet	71.60	-	10.23	8.83	5.40
Banana	50.14	2.40	1.19	1.18	1.01
DrugCLIP	79.45	47.82	37.86	30.76	10.10
DrugCLIP+sign	76.82	41.42	33.87	26.50	8.89
DrugHash	**83.73**	**57.16**	**43.03**	**37.18**	**12.07**

Table 1: Results on DUD-E

	AUROC	BEDROC	$EF^{0.5\%}$	$EF^{1\%}$	$EF^{5\%}$
Surflex	51.47	-	-	2.50	-
Glide-SP	53.15	4.00	3.17	3.41	2.01
Gnina	60.93	5.40	-	4.63	-
DeepDTA	56.27	2.53	-	1.47	-
Planet	57.31	-	4.64	3.87	2.43
Banana	**62.78**	5.02	3.98	3.79	**2.83**
DrugCLIP	56.36	6.78	7.77	5.66	2.32
DrugCLIP+sign	55.82	5.96	7.35	4.70	2.03
DrugHash	54.58	**7.14**	**9.65**	**6.14**	2.42

Table 2: Results on LIT-PCBA

the number of inactive compounds is 407,381. To evaluate the memory and time cost of different VS methods, we adopt the ZINC database (Irwin et al. 2020) and the Enamine REAL database (Shivanyuk et al. 2007). ZINC contains 2.3 million ready-to-dock molecules. Enamine REAL is a database designed for ultra-large scale VS which contains more than 6.75B molecules.

Baselines On the DUD-E dataset, we adopt 7 baselines, which include two docking methods Glide-SP (Friesner et al. 2004) and Vina (Trott and Olson 2010), and five learning based methods NN-score (Durrant and McCammon 2011), RF-Score (Ballester and Mitchell 2010), Pafnucy (Stepniewska-Dziubinska, Zielenkiewicz, and Siedlecki 2017), Planet (Zhang et al. 2023), Banana (Brocidiacono et al. 2022), and DrugCLIP (Gao et al. 2023). On the LIT-PCBA dataset, we adopt 7 baselines, which include three docking methods Surflex (Spitzer and Jain 2012), Glide-SP (Friesner et al. 2004) and Gnina (McNutt et al. 2021), and four learning based methods DeepDTA (Öztürk, Özgür, and Ozkirimli 2018), Planet (Zhang et al. 2023), Banana (Brocidiacono et al. 2022), and DrugCLIP (Gao et al. 2023). We directly apply $\mathrm{sign}(\cdot)$ function on the trained DrugCLIP model to obtain binary output as an additional baseline.

Implementation Details In our implementation, we set the hyperparameter λ to 0.2. The temperature coefficient τ is set to 0.07. Each time, we sample 48 protein-molecule pairs for contrastive learning. The code length of the output binary hash codes is 128. Our model is trained on NVIDIA RTX A6000 GPUs, and each model is trained up to 200 epochs. The model is trained for five random seeds and we report the average results. The CASF-2016 dataset (Su et al. 2018) is used as the validation set to select the best number of epoch. We utilize gradient accumulation, performing gradient backpropagation every four steps on a single GPU card, which is equivalent to using four cards for distributed training. The time test is running on the Intel Xeon Gold 6240R CPUs.

Results

Evaluation on DUD-E We compare the AUROC, BEDORC and EF of DrugHash with baselines on the DUD-E dataset. The results are shown in Table 1. DrugCLIP has several versions in the original paper. For a fair comparison, we train DrugCLIP using the same dataset and evaluate the zero-shot results on DUD-E, and the results are consistent with those of the original paper. DrugHash also does not perform any finetuning on the DUD-E dataset and the zero-shot results are reported. We can find that DrugHash outperforms the baselines across all metrics. DrugHash uses the same encoder and training data as DrugCLIP, but it outperforms DrugCLIP by a large margin across all metrics, which shows the superiority of our hashing strategy. When directly post-processing the output of DrugCLIP into binary values, the accuracy drops. This demonstrates the effectiveness of our hashing strategy which integrates the binary coding process into the whole learning process.

Evaluation on LIT-PCBA We compare the AUROC, BEDORC and EF of DrugHash with baselines on the LIT-PCBA dataset. The results are shown in Table 2. LIT-PCBA is a more challenging dataset compared to DUD-E, and we can find that the performance of all methods has declined. DrugHash outperforms all other methods on metrics of BEDROC, $EF^{05\%}$ and $EF^{1\%}$. The performance of DrugHash on AUROC is not outstanding. However, as pointed out in (Truchon and Bayly 2007), AUROC is not sensitive to early recognition. Early recognition means a good VS method should rank actives very early among all molecules, because only a small proportion of potential actives will be tested in experiments. Hence, AUROC is less meaningful compared with other metrics. Although DrugHash does not surpass Banana on $EF^{5\%}$, it significantly outperforms Banana on $EF^{05\%}$ and $EF^{1\%}$. $EF^{05\%}$ and $EF^{1\%}$ are more aligned with the requirement of early recognition than $EF^{5\%}$. Moreover, Banana generalizes poorly on the DUD-E dataset. Overall, the experiment results show the effectiveness of DrugHash.

Memory and Time cost We show the memory and time cost of Planet, Banana, DrugCLIP and DrugHash when adopting ZINC and Enamine REAL as the target molecule library in Table 3. Unlike the learning-based methods, Vina could only store the raw molecule files. The memory cost of raw molecule files is extremely high, so we do not present them in the table. For the ZINC database, which contains around 2.3 million ready-to-dock molecule files, the memory cost of real-valued vectors is still acceptable for the three real-valued methods. However, when the database size increases to the size of the REAL database, the memory cost for real-valued vectors rises to around 3080GB, which is

Model	Memory Cost		Loading | Computing Time Cost	
	ZINC (∼2.3M)	REAL (∼6.5B)	ZINC (∼2.3M)	REAL (∼6.5B)
Vina	-	-	- | 1863 days	- | -
Planet	2.57GB	7264GB	1.42 | 1749 seconds	1.20 | 1408 hours
Banana	1.00GB	3080GB	0.66 | 67.05 seconds	0.52 | 52 hours
DrugCLIP	1.00GB	3080GB	0.66 | 0.35 seconds	1876 | 1064 seconds
DrugHash	0.03GB	96GB	0.02 | 0.20 seconds	57 | 301 seconds

Table 3: Comparison of memory and time cost

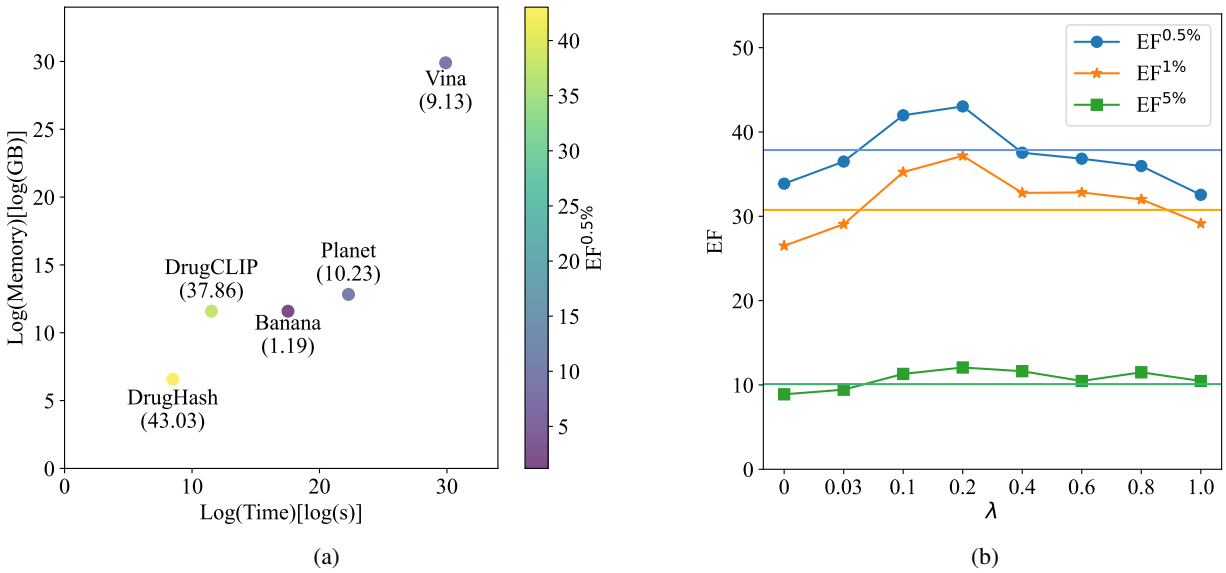

Figure 3: (a) Comprehensive comparison of $EF^{0.5\%}$, memory cost and time cost. (b) The results of $EF^{0.5\%}$, $EF^{1\%}$ and $EF^{5\%}$ on different λ.

unacceptable for the memory of most computers. However, DrugHash only needs 96GB to store the REAL database, which is about 32 times smaller than that of Banana and DrugCLIP, and 75 times smaller than that of Planet.

We also evaluate the time cost of the above methods when using only one target protein as the query. We show the data-loading time from storage and the computing time separately in Table 3. The time cost of Vina to dock the total ZINC dataset takes around 1863 days which is impractical in real-world applications, so we do not show the time cost for the larger REAL database. Though Planet and Banana can pre-encode the proteins and molecules to vector representations, they have to further fuse the protein and molecule representations to predict the final binding affinity or interaction. On a database with the same size of ZINC, their required time is still acceptable, but on REAL, their time cost rises to tens or even thousands of hours. DrugCLIP takes much less time on both databases compare with the above methods. DrugHash is the fastest method and achieves more than 4.6 times faster than DrugCLIP on the REAL database. DrugHash has more significant advantage as the number of target proteins increases.

We show the results of a comprehensive comparison of $EF^{0.5\%}$, memory cost and time cost in Figure 3a. We can see that Vina is not practical due to the large memory and time cost. Planet and Banana require less memory and time cost, but the cost is still too large for real applications, and their accuracy is not satisfactory. Although DrugCLIP performs VS at a relative faster speed, its pre-encoded molecular vectors still poses storage challenge for computational devices. DrugHash addresses the issue of high memory consumption required by the other learning-based methods and simultaneously enhances retrieval speed. Furthermore, DrugHash achieves the highest accuracy, which shows the promising potential of DrugHash in VS.

Analysis of Hashing Strategy

In this subsection, we will conduct a detailed analysis of how the hashing strategy works.

Sensitivity Analysis of λ We study the sensitivity of the hyperparameter λ in DrugHash. The results are shown in

Model	AUROC	BEDROC	$EF^{0.5\%}$	$EF^{1\%}$	$EF^{5\%}$
DrugHash-64	83.17	55.12	41.01	35.40	11.73
DrugHash-128 $\star$	83.73	57.16	43.03	37.18	12.07
DrugHash-256	84.96	59.23	43.67	37.76	12.50
DrugHash-512	84.06	60.50	44.44	39.30	12.57

Table 4: Results of DrugHash with different code length

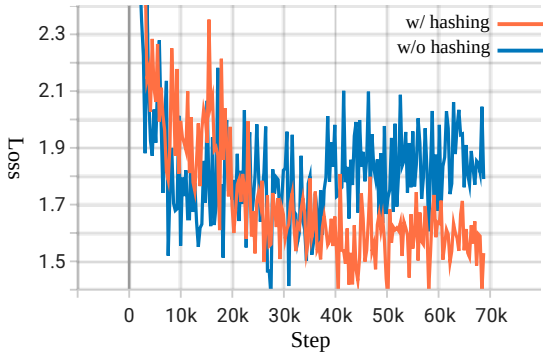

Figure 4: The loss curve on validation set.

Figure 3b. The horizontal line indicates the results obtained by removing the hashing strategy in training and directly using the real-valued vectors for calculation. Setting $\lambda = 0$ implies that no hashing strategy is applied during training; however, during evaluation, the $\mathrm{sign}(\cdot)$ function is still used to generate binary hash codes. This serves as an ablation study of the hashing strategy. In this case, we can find that the accuracy drops by a large margin, which proves the necessity of our hashing strategy. When λ is tuned from 0 to 1.0, the accuracy of DrugHash initially increases, peaking at $\lambda = 0.2$, before declining. When λ lies within the range of 0.1 and 0.4, the accuracy of binary retrieval can consistently exceed that of real-valued vectors. This demonstrates that our methods perform robustly within a relatively broad range of λ, and simply choosing λ within this range can yield better results compared to DrugClip.

Code Length Experiment We study the influence of the length of the binary hash codes. The results are shown in Table 4. DrugHash-128 is marked with $\star$ to denote our default implementation. We study the code length of $\{64, 128, 256, 512\}$, and the accuracy of DrugHash continuously improves as the code length increases. Therefore, the accuracy of DrugHash could be improved by simply increasing code length. However, a larger code length will bring larger memory and time cost. But due to the efficiency of binary hash codes, DrugHash is capable of adopting larger code lengths. Even with the code length of 512, DrugHash requires less than 200GB of memory cost to store the REAL database. DrugHash offers a good trade-off between accuracy and cost in real-world applications.

Overfitting Analysis We provide an explanation of how the hashing strategy enhances model accuracy. We demonstrate the loss curves of DrugHash on the validation set with and without the hashing strategy in Figure 4. In VS, the test space is much larger than the training/validation space and hence overfitting training/validation set is hard to avoid. In particular, the training/validation set consists of paired protein and molecule, but during testing (real deployment), each protein has to screen against tens of thousands of molecules. We can find that the model without the hashing strategy has the trend to overfit after 35k steps, with the loss on the validation set continuing to increase. However, after adding the hashing strategy, the loss on our validation set continues to decrease and reaches a lower value compared to the variant without hashing. The experiment indicates that the hashing strategy could alleviate the model overfitting and improve the accuracy.

Model	$EF^{5\%}$	$EF^{1\%}$	$EF^{0.5\%}$	Computing Time Cost
DrugHash	43.03	37.18	12.07	0.20 seconds
DrugHash+Faiss	42.00	37.08	12.08	0.06 seconds

Table 5: Performance of DrugHash with Faiss

Combined with Faiss

The searching speed of DrugHash can be further improved with Faiss (Douze et al. 2024). We denote the number of samples as N and configure the settings as follows: #cells $= 4\sqrt{N}$ and #probes $= 0.1 \times$ #cells. The index is constructed using IndexFlatIP. After embedding both the query protein and the molecule database with DrugHash, we employ Faiss to search for matching molecules. The accuracy on DUD-E dataset and the search time on the ZINC database are shown in Table 5. We can find that DrugHash can be combined with Faiss to further improve the speed with only a minimal loss in accuracy.

Conclusion

In this paper, we propose a hashing-based contrastive learning method, called DrugHash, for VS. DrugHash treats VS as a retrieval task that uses binary hash codes for efficient retrieval. In particular, DrugHash designs a simple yet effective hashing strategy to enable end-to-end learning of binary hash codes for both protein and molecule modalities, which can dramatically reduce the memory and time cost while improving accuracy. Experimental results show that DrugHash can outperform existing methods to achieve state-of-the-art accuracy, with at least a 32$\times$ reduction in memory cost and a 4.6$\times$ improvement in speed. We also conduct detailed experiments on how the hashing strategy works.

Acknowledgments

This work is supported by Key R&D Project of Jiangsu Province (No.BE2023652) and NSFC Project (No.12326615, No.62192783).

References

Ballester, P. J.; and Mitchell, J. B. 2010. A machine learning approach to predicting protein–ligand binding affinity with applications to molecular docking. *Bioinformatics*, 26(9): 1169–1175.

Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov, I. N.; and Bourne, P. E. 2000. The protein data bank. *Nucleic Acids Research*, 28(1): 235–242.

Brocidiacono, M.; Francoeur, P.; Aggarwal, R.; Popov, K. I.; Koes, D. R.; and Tropsha, A. 2022. BigBind: Learning from Nonstructural Data for Structure-Based Virtual Screening. *Journal of Chemical Information and Modeling*.

Bronstein, M. M.; Bronstein, A. M.; Michel, F.; and Paragios, N. 2010. Data fusion through cross-modality metric learning using similarity-sensitive hashing. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Chen, L.; Tan, X.; Wang, D.; Zhong, F.; Liu, X.; Yang, T.; Luo, X.; Chen, K.; Jiang, H.; and Zheng, M. 2020. TransformerCPI: improving compound–protein interaction prediction by sequence-based deep learning with self-attention mechanism and label reversal experiments. *Bioinformatics*, 36(16): 4406–4414.

Deng, C.; Chen, Z.; Liu, X.; Gao, X.; and Tao, D. 2018. Triplet-based deep hashing network for cross-modal retrieval. *IEEE Transactions on Image Processing*, 27(8): 3893–3903.

Douze, M.; Guzhva, A.; Deng, C.; Johnson, J.; Szilvasy, G.; Mazaré, P.-E.; Lomeli, M.; Hosseini, L.; and Jégou, H. 2024. The faiss library. *arXiv preprint arXiv:2401.08281*.

Durrant, J. D.; and McCammon, J. A. 2011. NNScore 2.0: a neural-network receptor–ligand scoring function. *Journal of Chemical Information and Modeling*, 51(11): 2897–2903.

Friesner, R. A.; Banks, J. L.; Murphy, R. B.; Halgren, T. A.; Klicic, J. J.; Mainz, D. T.; Repasky, M. P.; Knoll, E. H.; Shelley, M.; Perry, J. K.; et al. 2004. Glide: a new approach for rapid, accurate docking and scoring. 1. Method and assessment of docking accuracy. *Journal of Medicinal Chemistry*, 47(7): 1739–1749.

Gao, B.; Qiang, B.; Tan, H.; Jia, Y.; Ren, M.; Lu, M.; Liu, J.; Ma, W.-Y.; and Lan, Y. 2023. DrugCLIP: Contrasive Protein-Molecule Representation Learning for Virtual Screening. *Advances in Neural Information Processing Systems*.

Gaulton, A.; Bellis, L. J.; Bento, A. P.; Chambers, J.; Davies, M.; Hersey, A.; Light, Y.; McGlinchey, S.; Michalovich, D.; Al-Lazikani, B.; et al. 2012. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Research*, 40(D1): D1100–D1107.

Gorgulla, C.; Boeszoermenyi, A.; Wang, Z.-F.; Fischer, P. D.; Coote, P. W.; Padmanabha Das, K. M.; Malets, Y. S.; Radchenko, D. S.; Moroz, Y. S.; Scott, D. A.; et al. 2020. An open-source drug discovery platform enables ultra-large virtual screens. *Nature*, 580(7805): 663–668.

Hu, P.; Zhu, H.; Lin, J.; Peng, D.; Zhao, Y.-P.; and Peng, X. 2022. Unsupervised contrastive cross-modal hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3): 3877–3889.

Irwin, J. J.; and Shoichet, B. K. 2005. ZINC- a free database of commercially available compounds for virtual screening. *Journal of Chemical Information and Modeling*, 45(1): 177–182.

Irwin, J. J.; Tang, K. G.; Young, J.; Dandarchuluun, C.; Wong, B. R.; Khurelbaatar, M.; Moroz, Y. S.; Mayfield, J.; and Sayle, R. A. 2020. ZINC20—a free ultralarge-scale chemical database for ligand discovery. *Journal of Chemical Information and Modeling*, 60(12): 6065–6073.

Jiang, Q.-Y.; and Li, W.-J. 2017. Deep cross-modal hashing. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Jin, Z.; Wu, T.; Chen, T.; Pan, D.; Wang, X.; Xie, J.; Quan, L.; and Lyu, Q. 2023. CAPLA: improved prediction of protein–ligand binding affinity by a deep learning approach based on a cross-attention mechanism. *Bioinformatics*, 39(2): btad049.

Jones, D.; Kim, H.; Zhang, X.; Zemla, A.; Stevenson, G.; Bennett, W. D.; Kirshner, D.; Wong, S. E.; Lightstone, F. C.; and Allen, J. E. 2021. Improved protein–ligand binding affinity prediction with structure-based deep fusion inference. *Journal of Chemical Information and Modeling*, 61(4): 1583–1592.

Kitchen, D. B.; Decornez, H.; Furr, J. R.; and Bajorath, J. 2004. Docking and scoring in virtual screening for drug discovery: methods and applications. *Nature Reviews Drug Discovery*, 3(11): 935–949.

Kumar, S.; and Udupa, R. 2011. Learning hash functions for cross-view similarity search. In *International Joint Conference on Artificial Intelligence*.

Lin, Z.; Ding, G.; Hu, M.; and Wang, J. 2015. Semantics-preserving hashing for cross-view retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Lyu, J.; Wang, S.; Balius, T. E.; Singh, I.; Levit, A.; Moroz, Y. S.; O'Meara, M. J.; Che, T.; Algaa, E.; Tolmachova, K.; et al. 2019. Ultra-large library docking for discovering new chemotypes. *Nature*, 566(7743): 224–229.

McNutt, A. T.; Francoeur, P.; Aggarwal, R.; Masuda, T.; Meli, R.; Ragoza, M.; Sunseri, J.; and Koes, D. R. 2021. GNINA 1.0: molecular docking with deep learning. *Journal of Cheminformatics*, 13(1): 43.

Mysinger, M. M.; Carchia, M.; Irwin, J. J.; and Shoichet, B. K. 2012. Directory of useful decoys, enhanced (DUD-E): better ligands and decoys for better benchmarking. *Journal of Medicinal Chemistry*, 55(14): 6582–6594.

Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Öztürk, H.; Özgür, A.; and Ozkirimli, E. 2018. DeepDTA: deep drug–target binding affinity prediction. *Bioinformatics*, 34(17): i821–i829.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*.

Rarey, M.; Kramer, B.; Lengauer, T.; and Klebe, G. 1996. A fast flexible docking method using an incremental construction algorithm. *Journal of Molecular Biology*, 261(3): 470–489.

Raval, K.; and Ganatra, T. 2022. Basics, types and applications of molecular docking: A review. *IP International Journal of Comprehensive and Advanced Pharmacology*, 7(1): 12–16.

Shen, H. T.; Liu, L.; Yang, Y.; Xu, X.; Huang, Z.; Shen, F.; and Hong, R. 2020. Exploiting subspace relation in semantic labels for cross-modal hashing. *IEEE Transactions on Knowledge and Data Engineering*, 33(10): 3351–3365.

Shivanyuk, A. N.; Ryabukhin, S. V.; Tolmachev, A.; Bogolyubsky, A.; Mykytenko, D.; Chupryna, A.; Heilman, W.; and Kostyuk, A. 2007. Enamine real database: Making chemical diversity real. *Chemistry Today*, 25(6): 58–59.

Shuaibi, M.; Kolluru, A.; Das, A.; Grover, A.; Sriram, A.; Ulissi, Z.; and Zitnick, C. L. 2021. Rotation invariant graph neural networks using spin convolutions. *arXiv preprint arXiv:2106.09575*.

Spitzer, R.; and Jain, A. N. 2012. Surflex-Dock: Docking benchmarks and real-world application. *Journal of Computer-Aided Molecular Design*, 26: 687–699.

Stepniewska-Dziubinska, M. M.; Zielenkiewicz, P.; and Siedlecki, P. 2017. Pafnucy-a deep neural network for structure-based drug discovery. *Stat*, 1050: 19.

Su, M.; Yang, Q.; Du, Y.; Feng, G.; Liu, Z.; Li, Y.; and Wang, R. 2018. Comparative assessment of scoring functions: the CASF-2016 update. *Journal of Chemical Information and Modeling*, 59(2): 895–913.

Tran-Nguyen, V.-K.; Jacquemard, C.; and Rognan, D. 2020. LIT-PCBA: an unbiased data set for machine learning and virtual screening. *Journal of Chemical Information and Modeling*, 60(9): 4263–4273.

Trott, O.; and Olson, A. J. 2010. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of Computational Chemistry*, 31(2): 455–461.

Truchon, J.-F.; and Bayly, C. I. 2007. Evaluating virtual screening methods: good and bad metrics for the "early recognition" problem. *Journal of Chemical Information and Modeling*, 47(2): 488–508.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Wang, R.; Fang, X.; Lu, Y.; Yang, C.-Y.; and Wang, S. 2005. The PDBbind database: methodologies and updates. *Journal of Medicinal Chemistry*, 48(12): 4111–4119.

Wu, Y.; Gao, M.; Zeng, M.; Zhang, J.; and Li, M. 2022. BridgeDPI: a novel graph neural network for predicting drug–protein interactions. *Bioinformatics*, 38(9): 2571–2578.

Yang, J.; Roy, A.; and Zhang, Y. 2012. BioLiP: a semi-manually curated database for biologically relevant ligand–protein interactions. *Nucleic Acids Research*, 41(D1): D1096–D1103.

Yazdani-Jahromi, M.; Yousefi, N.; Tayebi, A.; Kolanthai, E.; Neal, C. J.; Seal, S.; and Garibay, O. O. 2022. AttentionSiteDTI: an interpretable graph-based model for drug-target interaction prediction using NLP sentence-level relation classification. *Briefings in Bioinformatics*, 23(4): bbac272.

Zhang, X.; Gao, H.; Wang, H.; Chen, Z.; Zhang, Z.; Chen, X.; Li, Y.; Qi, Y.; and Wang, R. 2023. Planet: a multi-objective graph neural network model for protein–ligand binding affinity prediction. *Journal of Chemical Information and Modeling*.

Zheng, L.; Fan, J.; and Mu, Y. 2019. Onionnet: a multiple-layer intermolecular-contact-based convolutional neural network for protein–ligand binding affinity prediction. *ACS Omega*, 4(14): 15956–15965.

Zhou, G.; Gao, Z.; Ding, Q.; Zheng, H.; Xu, H.; Wei, Z.; Zhang, L.; and Ke, G. 2023. Uni-mol: A universal 3d molecular representation learning framework. *International Conference on Learning Representations*.