MoleHD: Efficient Drug Discovery using Brain Inspired Hyperdimensional Computing

***Abstract*—Recently, machine learning algorithms have shown promising results in virtual screening of automated drug discovery by predicting molecular properties. While emerging learning methods such as graph neural networks and recurrent neural networks exhibit high accuracy, they are also notoriously computation-intensive and memory-intensive with operations such as feature embeddings or deep convolutions. In this paper, we propose `MoleHD`, an efficient learning model based on brain-inspired hyperdimensional computing (HDC) for molecular property prediction. We develop HDC encoders to project SMILES representation of a molecule into high-dimensional vectors that are used for HDC training and inference. We perform an extensive evaluation using 29 classification tasks from 3 widely-used molecule datasets (Clintox, BBBP, SIDER) under three splits methods (random, scaffold, and stratified). By a comprehensive comparison with 8 existing learning models including SOTA graph neural networks (GNNs), we show that `MoleHD` is able to achieve highest ROC-AUC score on random and scaffold splits on average across 3 datasets and achieve second-highest on stratified split. More importantly, `MoleHD` achieves such performance with significantly reduced computing cost than GNNs: no back-propagation needed, only around 10 minutes training time using CPU. The promising results presented in this paper can potentially lead to a novel path in drug discovery research. `MoleHD` is open-sourced and available at https://anonymous.4open.science/r/MoleHD-FDB5/.**

I. Introduction

Drug discovery is the process of using multi-disciplinary knowledge such as biology, chemistry and pharmacology to discover proficient medications amongst candidates according to safety and efficacy requirements. Modern drug discovery often features a cost-ineffective virtual screening process to select candidates from general chemical databases such as *ChEMBL* [4] and *OpenChem* [11] with large volume of molecular data to build a significant smaller in-house database for further synthesis. Therefore, data-driven machine learning techniques are increasingly applied into drug discovery, particularly in predicting molecular properties with drug discovery objectives.

Traditional machine learning algorithms such as random forest [7], support vector machine [16], k nearest neighbors [1], and gradient boosting [30] have been investigated in drug discovery applications. Such algorithms use molecular representations as input to predict molecular properties. However, because of limited sophistication, deep and complex structural information within a molecule is generally overlooked by those models. Thus, they typically do not exhibit strong capability in learning the features and only achieve sub-par performance. On the other hand, inspired by the recent success from other applications such as computer vision, neural network models have been increasingly applied in drug discovery. GNN learns representations by aggregating nodes and neighbouring information for molecular property predictions under different drug discovery objectives. However, molecular graphs often requires pre-processing or featurization. Extended-connectivity fingerprints (ECFP) is one of the most common featurization method that converts molecular graphs into fixed length representations, or fingerprints [26]. Such featurization algorithms usually requires comprehensive engineering efforts using chemical tool-chains such as the RDKit [14].

This paper takes a radical departure from common machine learning methods including neural networks by developing a low-cost brain-inspired hyperdimensional computing (HDC) model that requires less pre-processing efforts and is easier to implement. Inspired by the attributes of brain circuits including high-dimensionality and fully distributed holographic representation, this emerging computing paradigm postulates the generation, manipulation, and comparison of symbols represented by high dimensional vectors. Compared with DNNs, the advantages of HDC include smaller model size, less computation cost, and one/few-shot learning, making it a promising alternative computing paradigm [9]. Recently, HDC has demonstrated success on various application domains such as robotics [22], natural language processing [28], biomedical signal analysis [24], and biological sequence matching [6].

In this paper, we develop **`MoleHD`**, an HDC-based method to predict molecular properties in drug discovery. **`MoleHD`** first tokenizes SMILES strings of the input molecule into numerical list of tokens, and then apply HDC encoding to project realistic features into their high-dimensional space representations: hypervectors. Next, **`MoleHD`** leverages hypervector properties to train an HDC model that can be used to perform molecule classification tasks.

The qualitative advantages of **`MoleHD`** compared to existing neural network-based classifiers for drug discovery are listed as follows: 1). **back-propagation free**: **`MoleHD`** does not need back-propagation to train a set of parameters; instead, it uses one/few-shot learning to establish abstract patterns that can represent specific symbols. 2). **efficient computing**: unlike neural networks, **`MoleHD`** does not need complicated arithmetic operations such as convolutions which presents a major computing/energy burden to computing platforms; instead, it only uses simple arithmetic operations such as addition between two vectors. Thus, **`MoleHD`** only needs to run on commodity CPU and can finish both training and testing on the reported datasets within minutes, while GNN requires around 5 days for training using Nvidia GPU [29]. 3). **smaller model size**: **`MoleHD`** only needs to store a set of vectors for comparison during inference, while SOTA neural networks often need millions of parameters and requires memory in 100MB scale to store the parameters (e.g., weights and activation values) [19]. The main contributions of this paper are summarized as below:

- We propose **MoleHD**, an efficient novel learning model based on hyperdimensional computing. This promising results of **MoleHD** provide a viable option and alternative to existing learning methods in drug discovery domain. **MoleHD** is open-sourced and available to the public.
- We develop a complete molecular-specific pipeline for HDC-based drug discovery. **MoleHD** tokenizes SMILE strings into tokens representing the substructures and then project them into hypervectors during encoding. Then, **MoleHD** uses the encoded hypervectors to train and evaluate the classification model.
- We perform an extensive evaluation of **MoleHD** on 29 classification tasks from 3 widely-used molecule datasets under three splits methods. By a comprehensive comparison with 8 baseline models including SOTA neural networks, **MoleHD** is able to achieve highest ROC-AUC score on random and scaffold splits on average across 3 datasets and achieve second-highest on stratified split. More importantly, **MoleHD** achieves such performance with significantly reduced computing cost and smaller model size than GNNs.
- We conduct a design space exploration of **MoleHD** by evaluating two tokenization schemes (**MoleHD**-PE and **MoleHD**-char), three gram sizes (uni-gram and bi-gram), and evaluate their corresponding performance.

II. Related Works

A. Hyperdimensional Computing

Since HDC was proposed as a cognitive method by P. Kanerva [8], researchers have been focusing on developing diverse applications of HDC. Hyperdimensional computing (HDC), also known as vector-symbolic architectures (VSA), was introduced as an alternative computational model mimicking the "human brain" at the functionality level [8]. HDC has been used in modern robotics to perform active perception by integrating the sensory perceptions experienced by an agent with its motoric capabilities, which is vital to autonomous learning agents [22]. HDC has also been used in biomedical signal processing and exhibits 97.8% accuracy on hand gesture recognition based on EMG, which surpasses support vector machine by 8.1% [24]. Recent works also show that HDC outperforms other machine learning methods in DNA sequencing [6], [12]. HDC also out-performs conventional machine learning algorithms such as k-nearest neighbors and support vector machines with even more compact model in several natural language processing tasks [28], [18].

B. Drug Discovery with ML

Machine learning algorithms are used in drug discovery mostly in predicting molecular properties to determine if they satisfy the drug discovery objective. Earlier works focus on traditional ML algorithms such as logistic regression [30], random forest [7] and support vector machine [16]. However, because of model limitations, such traditional models generally achieve inferior performance. Recently, emerging machine learning algorithms such as GNNs are increasingly applied to drug discovery for achieving higher performance. GNNs leverages fingerprints derived from the molecular graph to learn the representations. Direct message passing neural network (D-MPNN) is an evolution of message passing neural networks that centers on bonds between atoms which is able to maintain two representations [31], [27]. Contrastive learning is also applied into GNNs to fuse drug discovery domain knowledge and molecular properties to augment learning of representations [3], [29]. In addition to GNNs, natural language processing (NLP) models such as recurrent neural networks (RNNs) are also introduced in drug discovery. Compared with GNNs, RNNs typically do not rely on complex fingerprint conversion process using toolchains such as **RDKit** [23], [17]. However, RNNs still require word embeddings tools such as **Smi2Vec**, to fully extract features from the molecule SMILES representation.

III. Preliminaries on HDC

A. Hypervectors

Hypervectors (HV) are high-dimensional (usually higher than 10,000), holographic (not micro-coded) vectors with (pseudo-)random and i.i.d. elements [8]. An HV with d dimensions can be denoted as $\vec{H} = \langle h_1, h_2, \ldots, h_d \rangle$, where h_i refers to the elements inside the HV. HVs are fundamental blocks in HDC that are able to accommodate and represent information in different scales and layers. When the dimensionality is sufficiently high (e.g., $D = 10,000$), any two random HVs are nearly orthogonal [8]. HDC utilizes different operations HVs support as means of producing aggregations of information or creating representations of new information.

B. Operations

In HDC, addition $(+)$, multiplication $(*)$ and permutation (ρ) are the three basic operations HVs can support. Additions and multiplications take two input HVs as operands and perform **element-wise** add or multiply operations on the two HVs. Permutation takes one HV as the input operand and perform **cyclic rotation** by a specific amount. All the operations do not modify the dimensionality of the input HVs, i.e. the input and the output HVs are in the same dimension. These three operations also have their corresponding physical meanings. Addition is used to aggregate same-type information, while multiplication is used to combine different types of information together to generate new information. Permutation is used to reflect spatial or temporal changes in the information, such as time series or spatial coordinates [8].

C. Similarity Measurement

In HDC, the similarity metric δ between the information that two HVs represent is measured by similarity check. Different algorithms can be used to calculate the similarity, such as the Euclidean ($L2$) distance, the Hamming distance (for binary HVs), and cosine similarity (which we use in this paper, as shown in Eq. 1). A higher similarity δ between two HVs shows that these two HVs have more information in common, or vice versa. Because of the high dimensionality of HVs,

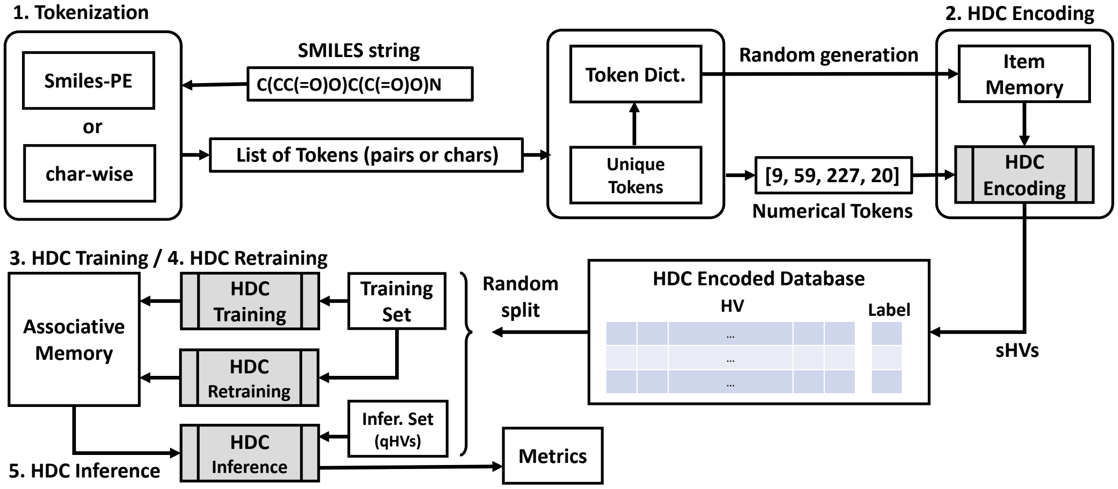


Fig. 1. Overview of **MoleHD**. **MoleHD** has 5 major steps: **Tokenization**, **Encoding**, **Training**, **Retraining** and **Inference**.

TABLE I
MoleHD-PE vs. **MoleHD**-char tokenization, using sulfate (O_4S^{-2}, CAS Registry Number: 14808-79-8) as an example.

input SMILES string	[O-]S(=O)(=O)[O-]
MoleHD-PE tokenization	'[O-]', 'S(=O)(=O)', '[O-]'
MoleHD-char tokenization	'[', 'O', '-', ']', 'S', '(', '=', 'O', ')', '(', '=', 'O', ')', '[', 'O', '-', ']'

addition generally results in a new HV that is approximately 50% similar to the two original HVs, while multiplication and permutation result in HVs that are orthogonal to the original HVs, i.e., not similar.

$$\delta(\vec{H_p}, \vec{H_q}) = \frac{\vec{H_p} \cdot \vec{H_q}}{||\vec{H_p}|| \times ||\vec{H_q}||} \tag{1}$$

IV. MoleHD Framework

In this section, we introduce the proposed **MoleHD** and how it utilizes HDC to perform learning tasks in drug discovery. An overview of **MoleHD** is presented in Fig. 1.

A. Tokenization

In **MoleHD**, tokenization is the process of converting molecule features into their corresponding set of numerical tokens. It basically consists of three procedures: converting the SMILES string into a list of tokens and then assign number for the tokens to obtain a list of numerical tokens which are ready for HDC processing.

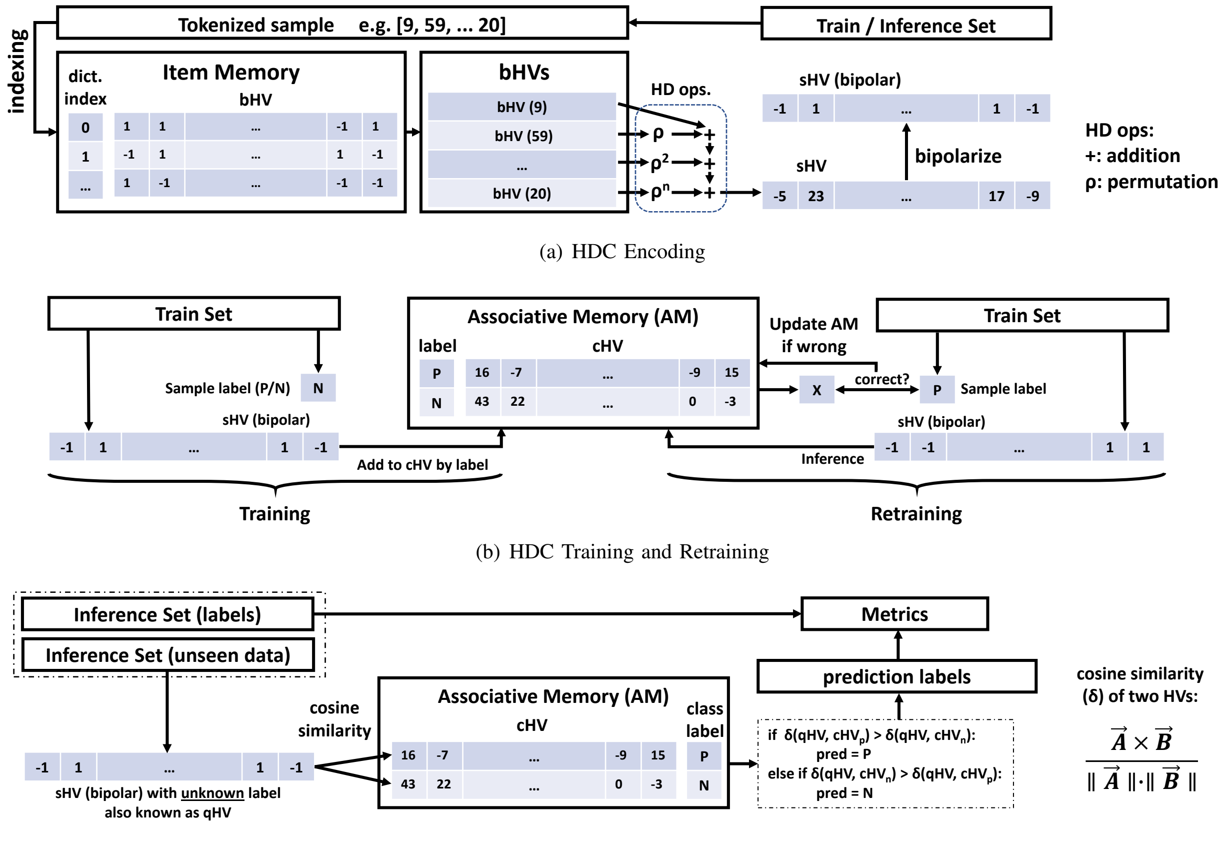
We develop two tokenization schemes for **MoleHD**: **MoleHD**-char and **MoleHD**-PE. **MoleHD**-char is the basic tokenization strategy that treats the input SMILES string as a textual string. **MoleHD**-char split the textual string into characters to obtain a list of tokens. Each unique character inside the string is then assigned with a unique random number to form the numerical tokens. **MoleHD**-PE uses the open-source SMILES Pair Encoding (SMILES-PE) model to extract the sub-structures in the input SMILES strings then assign a unique number based on their appearance frequency ranking to tokenize them. SMILES-PE is a data-driven algorithm to find substructures from a SMILES string [15]. **MoleHD**-PE uses SMILES-PE as-is as an add-on and does not require additional time for the pre-train. Due to model size limitation or other user-specific constraints, only m tokens will be stored. For missing tokens in **MoleHD**, a special token '0' is assigned. An example of the two strategies is shown in Table I.

B. HDC Encoding

Encoding is the process to project real-world features into their high-dimensional space representations: the HVs. In **MoleHD**, encoding process projects tokenized sample into its representing sample HV (sHV, or $\vec{S}$) via a combination of pre-defined HD operations as shown in Fig. 2(a).

Item Memory Item memory is generated from the token dictionary in tokenization. The item memory contains base HVs (bHV, or $\vec{B}$) in the same number ($m + 1$, considering the missing entry assigned as '0') as the entries in the token dictionary, i.e., each HV serves as the high-dimensional representation of a token. The item memory is fully random generated using a seed to ensure the i.i.d. properties. We note item memory as $\mathbb{B} = \{\vec{B_0}, \vec{B_1}, ..., \vec{B_m}\}$ where $\vec{B_i}$ is the base HV with index i.

HD operations in Encoding In **MoleHD**, encoding schemes can be flexible and data-specific. Algorithm 1 shows the process of uni-gram encoding as an example. Tokenized sample first uses its tokens iteratively in the item memory to index and fetch the corresponding base HVs. The base HVs permutate by their order in the tokenized sample and added up to establish the sample HV (Line 2 - 4). **MoleHD** also bipolarizes the elements inside the sample HV according to their relation with zero (Line 5 - 11). **MoleHD** also features bi-gram and tri-gram encoding which resembles the uni-gram encoding but



(a) HDC Encoding

(b) HDC Training and Retraining

(c) HDC Inference

Fig. 2. HDC processing: **Encoding**, **Training**, **Retraining** and **Inference**.

instead permutes every 2 or 3 tokens aggregated together by HV multiplication.

When all the available data are encoded, we can build an encoded database for all the corresponding sample HVs along with their labels. We then perform random split over the database to obtain the training set and inference set of HVs. We use $\vec{S_l}$ to represent a sample HV with label l.

```
Algorithm 1 MoleHD Encoding (uni-gram)
Input tokenized sample $T = \{t_0, t_1, t_k\}$, item memory $\mathbb{B}$.
Output sample HV $\vec{S}$
setZero($\vec{S}$)
for $t_i$ in $T$ do \* Perform uni-gram encoding. */
    $\vec{S} = \vec{S} + \rho^i(\vec{B_{t_i}})$
end for
for $s_i$ in $\vec{S}$ do \* Bipolarize the sample HV. */
    if $s_i > 0$ then $s_i = 1$
    else if $s_i < 0$ then $s_i = -1$
    end if
end for
```

C. HDC Training

Training is the process of establishing the associative memory $\mathbb{C} = \vec{C_1}, \vec{C_2}, ..., \vec{C_p}$ using the training set. Associative memory (AM) contains p class HVs (cHV, or $\vec{C}$), each representing a class in a learning task. Using a binary classification task as example shown in Fig. 2(b), AM contains the class HV representing positive ($\vec{C_P}$) and negative ($\vec{C_N}$). For each training sample, **MoleHD** adds its HV to the corresponding class HV according to the label, as shown in Eq. 3. This process is to aggregate the information from sample HVs together into the AM. However, one-epoch training is usually not enough to train a reliable AM for learning tasks, it is necessary to perform additional epochs for fine-tuning or retraining.

$$\vec{C_P} = \sum \vec{S_p}, \quad \vec{C_N} = \sum \vec{S_n} \tag{2}$$

D. HDC Retraining

Retraining is the process of fine-tuning the associative memory to enhance its accuracy using the training set, as shown in Fig. 2(b). For each training sample, **MoleHD** tries to use the AM to predict its label. If the prediction is correct, **MoleHD** proceeds to the next training sample. However, if the prediction is wrong, it indicates that the correct information of the sample HV has not been aggregated into the AM, or the information in the AM is not properly represented. Therefore, **MoleHD** performs an update to the AM to remove the erroneous and add the correct information, by subtracting the sample HV from the wrongly predicted class HV ($\vec{C_W}$) and adding it to the correct class HV ($\vec{C_R}$), as shown in Eq. 3.

$$\vec{C_W} = \vec{C_W} - \vec{S}, \quad \vec{C_R} = \vec{C_R} + \vec{S} \tag{3}$$

TABLE II
MoleHD CONFIGURATIONS AND (HYPER-)PARAMETERS

(hyper-)parameters	value
#tokens	1500
HV dimension	10000
gram size	uni-gram, bi-gram, tri-gram
epochs	150
splits	random, random stratified, scaffold
tokenization	SMILES-PE, char-wise

E. HDC Inference

Inference is the process of using unseen data from the inference set to evaluate the trained model's performance. As illustrated in Fig. 2(c), **MoleHD** calculates the cosine similarity (δ) between the sample HV from the inference set with unknown label (referred to as query HV (qHV, $\vec{Q_?}$) and each cHV in the AM to obtain the similarity values. The cHV with the most similarity indicates having the most overlap as to the preserved information with the qHV, i.e., class of inference sample, is subsequently predicted as $x = argmax(\delta(\vec{Q_?}, \mathbb{C}))$.

V. Experiments

A. Experimental Setup

Datasets We use 29 binary classification tasks in total from 3 datasets in the popular **MoleculeNet** benchmark suite for molecule machine learning [30]. For each dataset, we perform 0.8/0.2 random, stratified and scaffold split to build our training and inference set and repeat 5 experiments to get average performance as well as the score ranges. Details of the datasets are as follows:

- **BBBP** [20] contains 2052 drug compounds and their binary label (positive or negative) of permeability to the blood-brain barrier.
- **Clintox** [5] contains 1491 drug compounds and their binary label (positive or negative) of 1) clinical trial toxicity and 2) FDA approval status.
- **SIDER** [13] contains 1428 marketed drugs and their adverse drug reactions (ADR) in 27 individual tasks per **MedDRA** classifications [2]. Each task aims to classifying the positive (active) or negative (inactive) relationship between the drug compound and the ADR disorders of system organs.

Baseline Models We compare **MoleHD** with various baseline methods which are roughly in three categories: traditional learning models, GNNs and RNNs.

They can be classified into two groups. The first group features basic machine learning algorithms like logistic regression (LR), random forest (RF) and support vector machine (SVM). Basic learning algorithms are generally easier to implement, smaller in model size, and usually do not require extensive pre-processing for the inputs, although their capability may be limited. The second group features more advanced SOTA algorithms, including different graph convolutional neural network models, recurrent neural network models and hybrid models. Those models have stronger capabilities in classification, but require sophisticated data pre-processing such as embedding or vectorization, large model size, and higher memory footprints.

Details of the baselines compared in this paper are as follows:

- **Traditional learning Models** including logistic regression **(LR)**, random forest **(RF)**, and support vector machine **(SVM)** implemented and reported in the *MoleculeNet* benchmark [30] and *DeepChem* [25] framework.
- **Weave** [10], which is a graph convolution method that takes both local chemical environment and atom connectivity in featurization.
- **MolCLR** [29], which is a GNN with contrastive learning of representations with augmentations of atom masking, bond deletion, and subgraph removal.
- **D-MPNN** [31], [27], which is the directed message passing neural network that operates on molecular graphs.
- **LSTM** [23], which applies a modified version of the Smi2Vec tool to convert SMILE strings into atom vectors and then apply long short term memory (LSTM) RNN for the classification.
- **BiGRU** [17], which also uses Smi2Vec. It leverages the bidirectional gated recurrent unit (BiGRU) RNN to train sample vectors embedded in the atomic matrix.

Oversampling We perform oversampling on the smaller class of the training set by using naive random duplication, i.e., create multiple identical copies of samples from the smaller class to balance the dataset.

MoleHD Configurations We sweep a set of hyperparameters to obtain experimental results under different **MoleHD** configurations as listed according to Table II.

B. Metrics

Drug discovery datasets are mostly significantly imbalanced (e.g., the ratio of positive to negative samples can surpass 20:1), thus accuracy is generally not considered as a valid metric to reflect performance of a model. Receiver operating characteristics (ROC) curves and ROC Area-under-curve (AUC) scores are mostly embraced as the metric for model prediction performance, as suggested by benchmark datasets along with majority of literature [30], [25], [21].

Since HDC models are predicting using similarities, the "probability" used in calculating the ROC-AUC score requires specific definition. We propose two metrics for probability. First is to use "confidence level" (for being positive) η in Eq. 4. Confidence level is derived from similarities between query HV and the class HVs. The larger the difference, the higher the confidence of the HDC model prediction. Because the range of similarity difference is [-2, 2], to perform linear transformation to map the range of confidence level to [0, 1], we accordingly set 1/2 as the average value and 1/4 for coefficient of similarity difference, conforming with the probabilities. On the other hand, a Softmax functions can be applied directly over the similarities of each class which translates into the possibility required to calculate the ROC-AUC curve. Based on our implementation, we do not observe significant difference between these two methods.

TABLE III
MoleHD vs. Baselines on 3 datasets by average ROC-AUC score. Bold: the highest score. "-": data unavailable. Superscript "(k)": **MoleHD** ranks k-th place amongst all the available models under current dataset and split method.

split		random			stratified			scaffold	
dataset	Clintox	BBBP	SIDER	Clintox	BBBP	SIDER	Clintox	BBBP	SIDER
MoleHD tokenization gram size	**0.976**$^{(1)}$ char trigram	0.879$^{(3)}$ char trigram	0.599$^{(4)}$ PE unigram	0.973$^{(2)}$ char trigram	0.916$^{(3)}$ PE unigram	0.61$^{(2)}$ PE trigram	**0.987**$^{(1)}$ char bigram	0.844$^{(2)}$ char bigram	0.566$^{(4)}$ PE bigram
LR	0.733	0.737	0.643	-	0.728	-	-	0.699	-
RF	0.551	0.811	0.567	-	0.736	-	0.712	0.770	0.549
SVM	0.669	0.67	**0.656**	-	0.587	-	0.669	0.729	**0.682**
Weave	0.948	0.832	0.581	-	-	-	0.823	0.837	0.543
MolCLR	-	-	-	-	-	-	0.932	0.736	0.68
D-MPNN	0.892	**0.92**	0.639	0.898	0.932	**0.655**	0.874	**0.915**	0.606
LSTM	-	0.832	-	-	0.876	0.530	-	-	-
BiGRU	-	0.889	-	**0.978**	**0.946**	0.607	-	-	-

TABLE IV
MoleHD-PE performance on ROC-AUC score comparison on 3 datasets by average. Superscript and subscript refer to the upper and lower ranges. For SIDER dataset, the ROC-AUC score is task-average.

split		random			stratified			scaffold	
gram	uni-gram	bi-gram	tri-gram	uni-gram	bi-gram	tri-gram	uni-gram	bi-gram	tri-gram
Clintox	$0.941^{0.010}_{0.015}$	$0.897^{0.013}_{0.012}$	$0.881^{0.024}_{0.013}$	$0.960^{0.014}_{0.024}$	$0.970^{0.008}_{0.013}$	$0.932^{0.025}_{0.018}$	$0.952^{0.009}_{0.014}$	$0.966^{0.009}_{0.014}$	$0.930^{0.020}_{0.021}$
BBBP	$0.886^{0.014}_{0.024}$	$0.875^{0.012}_{0.021}$	$0.834^{0.021}_{0.021}$	$0.916^{0.014}_{0.014}$	$0.908^{0.016}_{0.025}$	$0.884^{0.016}_{0.026}$	$0.785^{0.024}_{0.023}$	$0.802^{0.021}_{0.021}$	$0.801^{0.020}_{0.018}$
SIDER	0.599	0.588	0.574	0.584	0.594	0.610	0.556	0.566	0.554

TABLE V
MoleHD-char performance on ROC-AUC score comparison on 3 datasets by average. Superscript and subscript refer to the upper and lower ranges. For SIDER dataset, the ROC-AUC score is task-average.

split		random			stratified			scaffold	
gram	uni-gram	bi-gram	tri-gram	uni-gram	bi-gram	tri-gram	uni-gram	bi-gram	tri-gram
Clintox	$0.955^{0.023}_{0.023}$	$0.971^{0.015}_{0.016}$	$0.976^{0.016}_{0.025}$	$0.956^{0.011}_{0.028}$	$0.971^{0.019}_{0.020}$	$0.973^{0.010}_{0.014}$	$0.966^{0.008}_{0.010}$	$0.987^{0.001}_{0.001}$	$0.982^{0.002}_{0.002}$
BBBP	$0.850^{0.022}_{0.028}$	$0.879^{0.034}_{0.029}$	$0.879^{0.026}_{0.020}$	$0.860^{0.017}_{0.020}$	$0.877^{0.014}_{0.013}$	$0.865^{0.012}_{0.015}$	$0.805^{0.009}_{0.011}$	$0.844^{0.006}_{0.010}$	$0.828^{0.004}_{0.002}$
SIDER	0.580	0.544	0.525	0.578	0.544	0.514	0.553	0.541	0.565

$$\eta = \frac{1}{2} + \frac{1}{4}(\delta(\vec{Q_?}, \vec{C_P}) - \delta(\vec{Q_?}, \vec{C_N})) \tag{4}$$

The experimental results are presented in two parts: the comparison between **MoleHD** and other baseline models as well as the comparison within **MoleHD** configurations.

C. *MoleHD vs. Baselines*

Most of the baseline models report results not as exhaustive as **MoleHD** in terms of split strategy. Therefore, we are performing comparison by "best effort", i.e., we compare best performing **MoleHD** with the baseline with data available under each split method of all the tasks. We are not able to report error bars or variations for this comparison because 1). many baseline models use inconsistent numbers of runs for average and/or cross-validations some of which are not reported, and, 2). some of the baselines just simply did not report any error bars or variation at all. However, for all the **MoleHD** versions we implemented, we report all the variations and score ranges of our method in Table IV and Table V. For results on **SIDER** dataset, the reported scores are task average and individual scores as well as corresponding range of each task are listed in Fig. 3 and Fig. 4.

We can observe from the results at Table III that, in general, **MoleHD** is achieving high ROC-AUC scores across datasets. For each split, **MoleHD** achieves a dataset-average ROC-AUC scores of **0.818**, **0.833** and **0.799** respectively, **ranking first on random and scaffold split and second on stratified split**, amongst the models with data available. Particularly, **MoleHD** performs greatly on the Clintox dataset particularly with scaffold split which are often regarded more challenging than the other splits where most of other baseline models are suffering from degradation, the score of **MoleHD** even increases instead.

We have an interesting observation that traditional models such as LR, RF and SVM exhibit poor performance over Clintox and BBBP datasets with significantly low score, however, they show competitive score for the SIDER dataset. On the contrary, while NNs usually performs greatly on Clintox and BBBP datasets, they only show sub-par score even lower than some traditional models, e.g., SVM achieves highest accuracy on SIDER dataset with random and scaffold split.

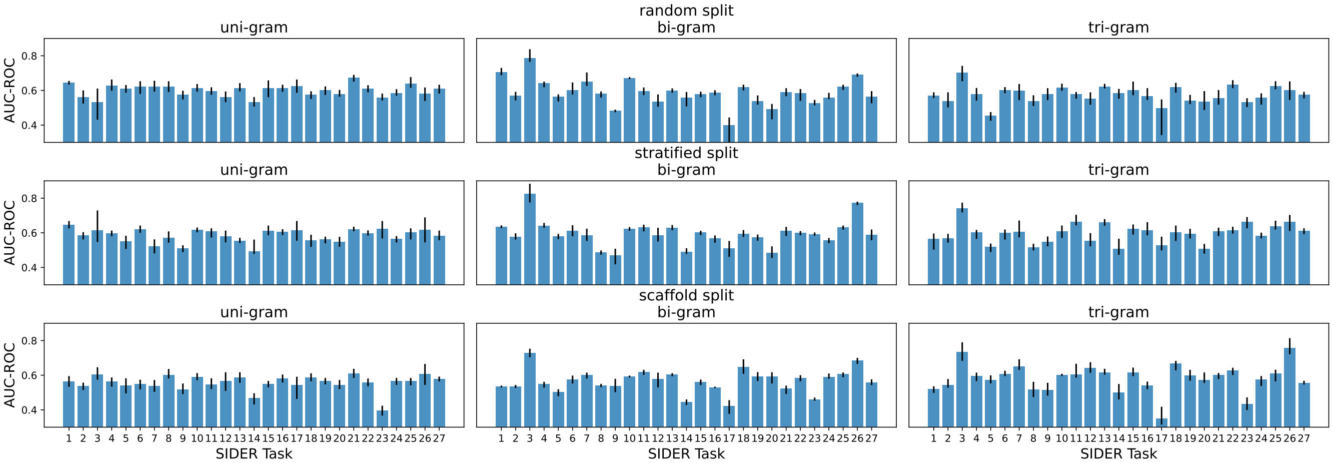


Fig. 3. **MoleHD**-PE performance comparison on all the 27 SIDER tasks under 3 split methods and 3 gram sizes.

Robustness-wise, **MoleHD** also outperforms other baseline models. For example, some GNN models are able to achieve top score on a specific dataset, however, they present much lower score on other datasets. For example, for **D-MPNN**, although it shows high scores at the BBBP dataset by ranking first at random and scaffold split, its score on the Clintox dataset seems mediocre. For Weave, it shows significantly degraded score on the Clintox dataset from random split to scaffold split. Such variation on performance would arouse questions on those models' transferability, while for **MoleHD**, the performance is largely consistent across different datasets and split methods.

D. *Comparisons within* ***MoleHD***

In addition to comparing with baseline models, we also evaluate an intensive set of **MoleHD** and dataset configurations, including: two different tokenization schemes (**MoleHD**-PE and **MoleHD**-char), three gram sizes (uni-gram, bi-gram and tri-gram), and three dataset split methods (random, stratified and scaffold split) as illustrated in Table IV and Table V.

We do not observe significant differences on the performance within all the configurations in general. The performance of **MoleHD** is overall consistent, thus, there is no single configuration that can dominate other configurations for most, if not all, the datasets and split methods. However, we do observe that for the scaffold split, although it is generally considered the harder split than random or stratified, the score variation is generally smaller than that of random and stratified split for **MoleHD**. For example, for task 3, significant variation can be observed under most configurations of random and stratified split while for scaffold split, the variation is much less. This can relate to the fundamentals behind the split methods that scaffold split leverages the structural information of the molecule when splitting which can help HDC encoding methods for extracting features.

E. *Efficiency of* ***MoleHD***

We elaborate the computing cost of **MoleHD** and compare it with SOTA neural networks. 1) Unlike GNNs, **MoleHD** does not require specific effort on pre-training the model. 2) For all the reported datasets, **MoleHD** is able to achieve the reported accuracy **within 10 minutes** using CPU only from the commodity desktop (AMD Ryzen™ 5 3600 3.6 GHz). Note that this includes both training and inference for each dataset. 3) **MoleHD** also requires less space for model storage as for one binary classification task, model size of **MoleHD** is only around 80kB and during run-time, the memory footprint of **MoleHD** is also generally less than 10MB. For GNNs as comparison, extensive pre-training can be necessary, e.g., MolCLR requires around 5 days of pre-training using Nvidia® Quadro RTX™ 6000 as reported in the corresponding literature [29]. GNNs and RNNs are also harder to implement considering the effort of establish multiple layers with considerable amount of nodes with the model size at 100MB level, especially considering the necessity of performing back-propagation (which is not needed in HDC) during training with millions of parameters in total [19].

VI. Conclusion

In this paper, we propose **MoleHD**, an efficient learning model which leverages the novel brain-inspired hyperdimensional computing for molecule property prediction in drug discovery applications. **MoleHD** projects SMILES strings of drug compound into hypervectors in the hyperdimensional space to extract features. The hypervectors are then used during training, retraining and inference of the HDC model to perform learning tasks. We evaluate **MoleHD** on 29 classification tasks from 3 widely-used benchmark datasets and compare **MoleHD** performance with 8 baseline machine learning models including SOTA GNNs and RNNS. According to experimental results, **MoleHD** is able to achieve highest ROC-AUC score on random and scaffold splits on average across 3 datasets and achieve second-highest on stratified split. Compared with traditional models and NNs, **MoleHD** also requires less training efforts, smaller model size, as well as smaller computation costs. This work marks the potential of using hyperdimensional computing as an alternative to the existing models in the drug discovery domain.

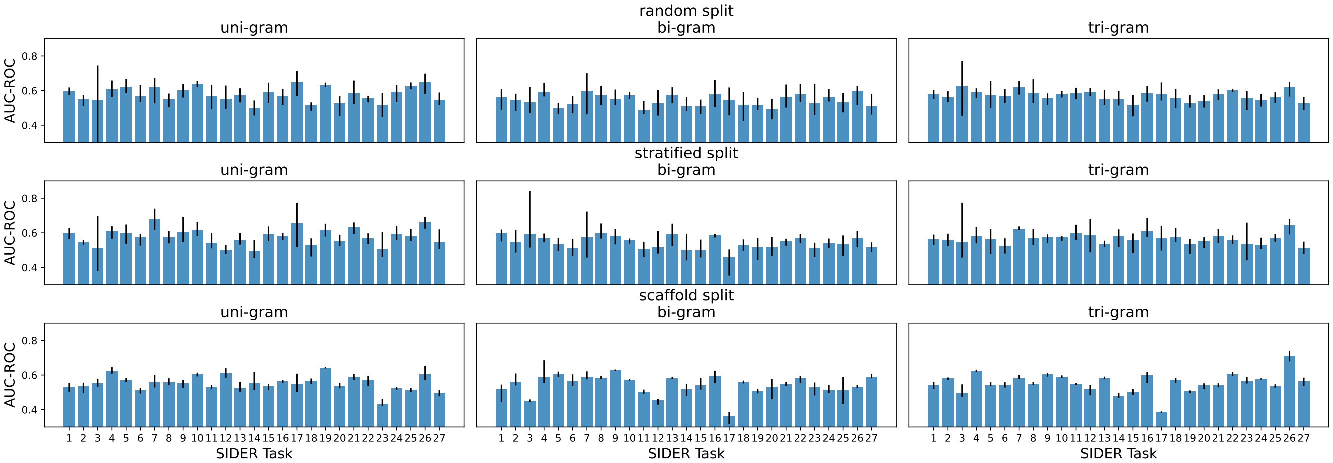


Fig. 4. **MoleHD**-char performance comparison on all the 27 SIDER tasks under 3 split methods and 3 gram sizes.

References

[1] Roya Arian, Amirali Hariri, Alireza Mehridehnavi, Afshin Fassihi, and Fahimeh Ghasemi. Protein kinase inhibitors' classification using k-nearest neighbor algorithm. *Computational biology and chemistry*, 86:107269, 2020.

[2] Elliot G Brown, Louise Wood, and Sue Wood. The medical dictionary for regulatory activities (meddra). *Drug safety*, 20(2):109–117, 1999.

[3] Yin Fang, Haihong Yang, Xiang Zhuang, Xin Shao, Xiaohui Fan, and Huajun Chen. Knowledge-aware contrastive molecular graph learning. *arXiv preprint arXiv:2103.13047*, 2021.

[4] Anna Gaulton, Louisa J Bellis, A Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, et al. Chembl: a large-scale bioactivity database for drug discovery. *Nucleic acids research*, 40(D1):D1100–D1107, 2012.

[5] Kaitlyn M Gayvert, Neel S Madhukar, and Olivier Elemento. A data-driven approach to predicting successes and failures of clinical trials. *Cell chemical biology*, 23(10):1294–1301, 2016.

[6] Mohsen Imani, Tarek Nassar, Abbas Rahimi, and Tajana Rosing. Hdna: Energy-efficient dna sequencing using hyperdimensional computing. In *2018 IEEE EMBS International Conference on Biomedical &amp; Health Informatics (BHI)*, pages 271–274. IEEE, 2018.

[7] PB Jayaraj, Mathias K Ajay, M Nufail, G Gopakumar, and UC Abdul Jaleel. Gpurfscreen: a gpu based virtual screening tool using random forest classifier. *Journal of cheminformatics*, 8(1):1–10, 2016.

[8] Pentti Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive computation*, 1(2):139–159, 2009.

[9] Geethan Karunaratne, Manuel Le Gallo, Giovanni Cherubini, Luca Benini, Abbas Rahimi, and Abu Sebastian. In-memory hyperdimensional computing. *Nature Electronics*, 3(6):327–337, 2020.

[10] Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design*, 30(8):595–608, 2016.

[11] Sunghwan Kim, Paul A Thiessen, Evan E Bolton, Jie Chen, Gang Fu, Asta Gindulyte, Lianyi Han, Jane He, Siqian He, Benjamin A Shoemaker, et al. Pubchem substance and compound databases. *Nucleic acids research*, 44(D1):D1202–D1213, 2016.

[12] Yeseong Kim, Mohsen Imani, Niema Moshiri, and Tajana Rosing. Geniehd: Efficient dna pattern matching accelerator using hyperdimensional computing. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 115–120. IEEE, 2020.

[13] Michael Kuhn, Ivica Letunic, Lars Juhl Jensen, and Peer Bork. The sider database of drugs and side effects. *Nucleic acids research*, 44(D1):D1075–D1079, 2016.

[14] Greg Landrum. Rdkit documentation. *Release*, 1(1-79):4, 2013.

[15] Xinhao Li and Denis Fourches. Smiles pair encoding: A data-driven substructure tokenization algorithm for deep learning. 2020.

[16] Chin Y Liew, Xiao H Ma, Xianghui Liu, and Chun W Yap. Svm model for virtual screening of lck inhibitors. *Journal of chemical information and modeling*, 49(4):877–885, 2009.

[17] Xuan Lin, Zhe Quan, Zhi-Jie Wang, Huang Huang, and Xiangxiang Zeng. A novel molecular representation with bigru neural networks for learning atom. *Briefings in bioinformatics*, 21(6):2099–2111, 2020.

[18] Fangxin Liu, Haomin Li, Xiaokang Yang, and Li Jiang. L3e-hd: A framework enabling efficient ensemble in high-dimensional space for language tasks. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1844–1848, 2022.

[19] Hehuan Ma, Yatao Bian, Yu Rong, Wenbing Huang, Tingyang Xu, Weiyang Xie, Geyan Ye, and Junzhou Huang. Multi-view graph neural networks for molecular property prediction. *arXiv preprint arXiv:2005.13607*, 2020.

[20] Ines Filipa Martins, Ana L Teixeira, Luis Pinheiro, and Andre O Falcao. A bayesian approach to in silico blood-brain barrier penetration modeling. *Journal of chemical information and modeling*, 52(6):1686–1697, 2012.

[21] Andreas Mayr, Günter Klambauer, Thomas Unterthiner, Marvin Steijaert, Jörg K Wegner, Hugo Ceulemans, Djork-Arné Clevert, and Sepp Hochreiter. Large-scale comparison of machine learning methods for drug target prediction on chembl. *Chemical science*, 9(24):5441–5451, 2018.

[22] Anton Mitrokhin, P Sutor, Cornelia Fermüller, and Yiannis Aloimonos. Learning sensorimotor control with neuromorphic sensors: Toward hyperdimensional active perception. *Science Robotics*, 4(30), 2019.

[23] Zhe Quan, Xuan Lin, Zhi-Jie Wang, Yan Liu, Fan Wang, and Kenli Li. A system for learning atoms based on long short-term memory recurrent neural networks. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 728–733. IEEE, 2018.

[24] Abbas Rahimi et al. Hyperdimensional biosignal processing: A case study for emg-based hand gesture recognition. In *ICRC*, 2016.

[25] Bharath Ramsundar, Peter Eastman, Patrick Walters, and Vijay Pande. *Deep learning for the life sciences: applying deep learning to genomics, microscopy, drug discovery, and more.* ” O’Reilly Media, Inc.”, 2019.

[26] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010.

[27] Kyle Swanson. *Message passing neural networks for molecular property prediction*. PhD thesis, Massachusetts Institute of Technology, 2019.

[28] Rahul Thapa, Bikal Lamichhane, Dongning Ma, and Xun Jiao. Spamhd: Memory-efficient text spam detection using brain-inspired hyperdimensional computing. In *2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 84–89. IEEE, 2021.

[29] Yuyang Wang, Jianren Wang, Zhonglin Cao, and Amir Barati Farimani. Molclr: Molecular contrastive learning of representations via graph neural networks. *arXiv preprint arXiv:2102.10056*, 2021.

[30] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[31] Kevin Yang et al. *Are learned molecular representations ready for prime time?* PhD thesis, Massachusetts Institute of Technology, 2019.