MoleHD: Efficient Drug Discovery using Brain Inspired Hyperdimensional Computing

Dongning Ma, Rahul Thapa, Xun Jiao
Villanova University
Villanova, PA 19085
{dma2, rthapa, xjiao}@villanova.edu

***Abstract*—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, we show that `MoleHD` achieves 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: no back-propagation needed, only around 10 minutes training time using CPU. `MoleHD` is open-sourced and available at https://github.com/VU-DETAIL/MoleHD.**

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 to select candidates from general databases such as *ChEMBL* and *OpenChem* 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 [1].

This paper makes a further step to develop a even lower-cost drug discovery scheme, **`MoleHD`**, based on the emerging brain inspired hyperdimensional computing (HDC) scheme. Inspired by the attributes of brain circuits including high-dimensionality and fully distributed holographic representation, HDC 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 in various application domains [2]–[4]. Compared with other machine learning algorithms, **`MoleHD`** requires less pre-processing efforts and is easier to implement. Specifically, **`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 main contributions of this paper are summarized as below:

- We develop **`MoleHD`**, an HDC-based molecular-specific pipeline for efficient drug discovery. **`MoleHD`** tokenizes SMILE strings into tokens representing the substructures and then project them into hypervectors. Then, **`MoleHD`** uses hypervectors to train and evaluate the classification model.
- We perform an evaluation of **`MoleHD`** on 29 classification tasks from 3 molecule datasets under 3 splits methods. Out of 8 baselines, **`MoleHD`** is able to achieve best ROC-AUC score on 2 of the 3 split methods on average. More importantly, **`MoleHD`** achieves such performance with significantly reduced computing cost and model size. We also conduct a comprehensive design space exploration of **`MoleHD`** performance by evaluating 2 tokenization schemes and 3 gram sizes.

II. Preliminaries on HDC

Hypervectors: Hypervectors (HV) are high-dimensional (usually higher than 10,000), holographic (not micro-coded) vectors with (pseudo-)random and i.i.d. elements [5]. 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. HDC utilizes different operations HVs support as means of producing aggregations of information or creating representations of new information.

Operations: In HDC, addition ($+$), multiplication ($*$) and permutation (ρ) are the three basic operations HVs can support. Additions and multiplications take two 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 since they are element-wise.

Similarity Measurement: In HDC, the similarity metric δ between the information that two HVs represent is measured by similarity. Different algorithms can be used to calculate the similarity, such as the Euclidean (L_2) distance, the Hamming distance (for binary HVs), and cosine similarity (which we use in this paper). A higher similarity δ between two HVs shows that these two HVs have more information in common.

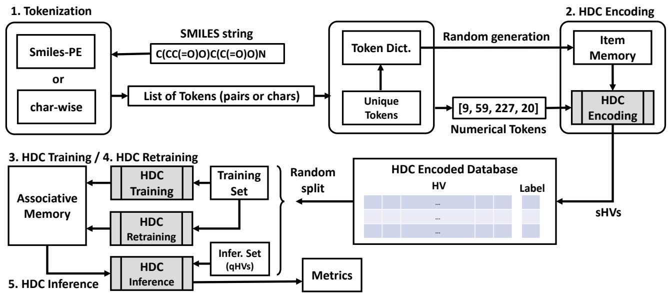


Fig. 1. Overview of **MoleHD**. **MoleHD** has 5 major steps: **Tokenization**, **Encoding**, **Training**, **Retraining** and **Inference**.

III. 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.

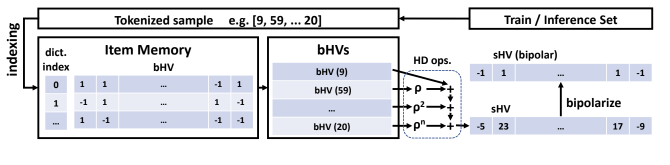
A. Tokenization

In **MoleHD**, tokenization is the process of converting molecule features into their corresponding set of numerical tokens. We develop two tokenization schemes: **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 [6]. **MoleHD**-PE uses SMILES-PE as-is and does not require additional pre-training.

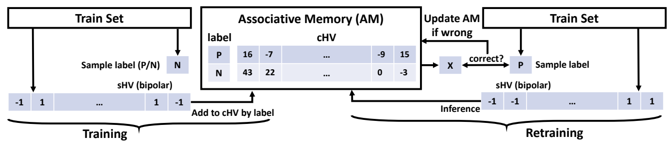
B. HDC Encoding

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).

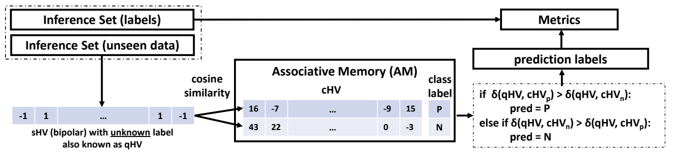
Item Memory: Item memory is generated from the token dictionary in tokenization. The item memory contains base HVs (bHV, or $\vec{B}$) in the same number ($m+1$, considering the missing entry assigned as '0') as the entries in the token dictionary, i.e., each HV serves as the high-dimensional representation of a token. The item memory is randomly 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.



(a) HDC Encoding



(b) HDC Training and Retraining


Inference Set (labels)
Inference Set (unseen data)
Metrics
prediction labels
cosine similarity
Associative Memory (AM)
cHV
class label
16 -7 ... -9 15 P
43 22 ... 0 -3 N
-1 1 ... 1 -1
sHV (bipolar) with unknown label also known as qHV
if δ(qHV, cHVp) > δ(qHV, cHVn): pred = P
else if δ(qHV, cHVn) > δ(qHV, cHVp): pred = N

(c) HDC Inference

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

HD operations in Encoding: In **MoleHD**, encoding schemes can be flexible and data-specific. 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. **MoleHD** also bipolarizes the elements inside the sample HV according to their relation with zero. **MoleHD** features bi-gram and tri-gram encoding which resembles the uni-gram encoding but 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.

C. HDC Training

Training is to build the associative memory using the training set. Associative memory (AM) $\mathbb{C} = \vec{C_1}, \vec{C_2}, ..., \vec{C_p}$ contains p class HVs (cHV, or $\vec{C}$), each representing a class. Using a binary classification task as example shown in Fig. 2(b), AM contains two class HVs, representing positive ($\vec{C_P}$) and negative ($\vec{C_N}$). For each training sample, **MoleHD** adds its HV to the corresponding class HV based on the label, i.e., $\vec{C_P} = \sum \vec{S_p}$ and $\vec{C_N} = \sum \vec{S_n}$. This process is to aggregate the information from sample HVs together into the AM.

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 correct, **MoleHD** proceeds to the next training sample. If wrong, it indicates 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 ($\vec{C_R}$), i.e., $\vec{C_W} = \vec{C_W} - \vec{S}$ and $\vec{C_R} = \vec{C_R} + \vec{S}$.

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 $x = argmax(\delta(\vec{Q_?}, \mathbb{C}))$.

IV. Experimental Results

A. Experimental Setup

Datasets: We use 29 binary classification tasks in total from 3 datasets: **BBBP** [7], **Clintox** [8] and **SIDER** [9]. 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.

Baselines: We compare **MoleHD** with 8 baselines: basic machine learning algorithms like logistic regression (LR), random forest (RF) and support vector machine (SVM) [1], [10] and more advanced SOTA algorithms, such as different graph convolutional neural network models including **Weave** [11], **MolCLR** [12] and **D-MPNN** [13], as well as recurrent neural network models including **LSTM** [14] and **BiGRU** [15].

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 [1], [10]. Since HDC models are predicting using similarities, the "probability" used in calculating the ROC-AUC score requires specific definition. We propose to use "confidence level" (for being positive) $\eta = \frac{1}{2} + \frac{1}{4}(\delta(\vec{Q_?}, \vec{C_P}) - \delta(\vec{Q_?}, \vec{C_N}))$ as the metric. 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.

*B. **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 II. For results on **SIDER** dataset, the reported scores are task average.

We can observe from the results at Table I 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.

Robustness-wise, **MoleHD** also outperforms other baseline models. For example, some GNN models can achieve top score on a specific dataset, however, they perform much worse on other datasets. For example, for **D-MPNN**, although it shows high scores at the BBBP 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.

*C. 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 II.

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**. 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.

*D. Efficiency of **MoleHD***

Unlike GNNs, **MoleHD** does not require any specific effort on pre-training the model while for GNNs as comparison, extensive pre-training can be necessary. **MoleHD** is able to achieve all the reported accuracy **within 10 minutes** using CPU only from the commodity desktop, including both training and inference from scratch. **MoleHD** also requires less space for model storage as for one binary classification task,

TABLE I
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	$\mathbf{0.976}^{(1)}$	$0.879^{(3)}$	$0.599^{(4)}$	$0.973^{(2)}$	$0.916^{(3)}$	$0.61^{(2)}$	$\mathbf{0.987}^{(1)}$	$0.844^{(2)}$	$0.566^{(4)}$
tokenization	char	char	PE	char	PE	PE	char	char	PE
gram size	trigram	trigram	unigram	trigram	unigram	trigram	bigram	bigram	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 II
MoleHD performance on ROC-AUC score comparison on 3 datasets. 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
char	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
PE	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

model size of **MoleHD** is only around 80kB and during run-time, the memory footprint is also generally less than 10MB while GNNs can have millions of parameters with the model size at 100MB level.

V. 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 which are then used during training, retraining and inference of the HDC model. We evaluate **MoleHD** on 29 classification tasks from 3 widely-used benchmark datasets and compare **MoleHD** performance with 8 baseline machine learning models. **MoleHD** is able to achieve highest ROC-AUC score on 2 out of 3 split methods. **MoleHD** also requires less training efforts, smaller model size, and lower computation costs. This work marks the potential of using hyperdimensional computing as an alternative efficient solution in the drug discovery domain.

References

[1] Z. Wu *et al.*, "Moleculenet: A benchmark for molecular machine learning," *Chemical science*, vol. 9, no. 2, pp. 513–530, 2018.

[2] R. Thapa *et al.*, "Spamhd: Memory-efficient text spam detection using brain-inspired hyperdimensional computing," in *2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, IEEE, 2021, pp. 84–89.

[3] A. Mitrokhin *et al.*, "Learning sensorimotor control with neuromorphic sensors: Toward hyperdimensional active perception," *Science Robotics*, vol. 4, no. 30, 2019.

[4] M. Imani *et al.*, "Hdna: Energy-efficient dna sequencing using hyperdimensional computing," in *2018 IEEE EMBS International Conference on Biomedical &amp; Health Informatics (BHI)*, IEEE, 2018, pp. 271–274.

[5] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognitive computation*, vol. 1, no. 2, pp. 139–159, 2009.

[6] X. Li and D. Fourches, "Smiles pair encoding: A data-driven substructure tokenization algorithm for deep learning," 2020.

[7] I. F. Martins *et al.*, "A bayesian approach to in silico blood-brain barrier penetration modeling," *Journal of chemical information and modeling*, vol. 52, no. 6, pp. 1686–1697, 2012.

[8] K. M. Gayvert *et al.*, "A data-driven approach to predicting successes and failures of clinical trials," *Cell chemical biology*, vol. 23, no. 10, pp. 1294–1301, 2016.

[9] M. Kuhn *et al.*, "The sider database of drugs and side effects," *Nucleic acids research*, vol. 44, no. D1, pp. D1075–D1079, 2016.

[10] B. Ramsundar *et al.*, *Deep learning for the life sciences: applying deep learning to genomics, microscopy, drug discovery, and more.* " O'Reilly Media, Inc.", 2019.

[11] S. Kearnes *et al.*, "Molecular graph convolutions: Moving beyond fingerprints," *Journal of computer-aided molecular design*, vol. 30, no. 8, pp. 595–608, 2016.

[12] Y. Wang *et al.*, "Molclr: Molecular contrastive learning of representations via graph neural networks," *arXiv preprint arXiv:2102.10056*, 2021.

[13] K. Swanson, "Message passing neural networks for molecular property prediction," Ph.D. dissertation, Massachusetts Institute of Technology, 2019.

[14] Z. Quan *et al.*, "A system for learning atoms based on long short-term memory recurrent neural networks," in *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, IEEE, 2018, pp. 728–733.

[15] X. Lin *et al.*, "A novel molecular representation with bigru neural networks for learning atom," *Briefings in bioinformatics*, vol. 21, no. 6, pp. 2099–2111, 2020.