N-Gram Graph, A Novel Molecule Representation

Shengchao Liu
University of Wisconsin-Madison
Department of Computer Science
shengchao@cs.wisc.edu

Thevaa Chandereng
University of Wisconsin-Madison
Department of Statistics
chandereng@wisc.edu

Yingyu Liang
University of Wisconsin-Madison
Department of Computer Science
yliang@cs.wisc.edu

Abstract

Recently, machine learning techniques have been widely applied in virtual screening for predicting the properties of molecules. It can provide strategies for prioritizing molecules for physical screens and significantly reduce resources for various applications in medicine, chemistry, and biology. Despite the increasing interest, the key challenging of constructing proper representations of molecules for the learning algorithms remains largely open. This paper introduces N-gram graph, a novel representation for molecules. It is simple, can be efficiently computed on general molecular graphs, and can be used with various machine learning methods. Experiments on several data sets demonstrate that the novel representation is able to reach the state-of-the-art performance on multiple tasks, even with simple machine learning models.

Introduction

The goal in drug discovery tasks is to test the properties of the molecules. Traditional physical screening is typically accurate and valid but also very costly and slow. In contrast, virtual screening using machine learning can be done in minutes for predictions on millions of molecules. Therefore, virtual screening can be a good filtering step before the physical experiment, so as to help accelerate the drug discovery process. To achieve this goal, the predictions generated by virtual screening should be made accurate.

The advances in deep learning have led to many achievements in the area of image classification and speech recognition, and recent efforts have applied to various other problems including, in particular, virtual screening. But unlike image and speech data, the most common raw input in drug discovery provide only highly abstract representations of the chemicals, which are not directly well-handled by existing learning systems. This creates a stumbling block in making good predictions on molecules. To address this issue, various representation methods are proposed, and the most commonly used ones are reviewed below.

Chemical fingerprints, perhaps the most widely used feature representations, encode each molecule as a fixed length bit vector. The prototypical method is the Morgan fingerprints, where each bit vector corresponds to a hashing bucket. To construct the fingerprint, first check the presence of a set of substructures, then hash the presented ones to the buckets, and finally set the bit based on whether the corresponding bucket is empty or not. Due to the hashing collisions, it is difficult to interpret such fingerprints and exam how the machine learning systems utilize them. Another prototypical method, Simplified Molecular Input Line Entry System (SMILES), is a character sequence

Preprint. Work in progress.

describing molecular structures. There are some inherent issues in SMILES, the biggest being that molecules cannot be simply represented as a linear sequence: the properties of drug-like organic molecules usually have dependence on ring structures and tree-like branching, whose information is lost in a linear sequence. An example of Morgan fingerprints and SMILES is illustrated in Figure 1.

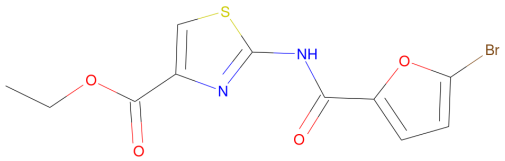


Figure 1: Illustration of three ways for molecule representation. The molecule graph is displayed on a 2D space. The corresponding canonical SMILES is c1cc(oc1C(=O)Nc2nc(cs2)C(=O)OCC)Br, and Morgan Fingerprints is, for example, [000000...00100100100...000000].

The benefits of applying deep neural networks in the horizon of drug discovery are yet to be fully realized, but we can make a conceptual analog to the image classification problems. Molecules can be simply visualized as a 3D graph, and we can easily fit images into this pattern: images are highly structured 2D graphs, where each pixel is the vertex in the graph and is connected to pixels in at most four directions (up, down, left, right). Thus this implicit attribute in images can filter out useful information after the convolution layers and pooling layers in the state-of-the-art deep convolution models. Following this idea, graph-structure data, including social networks and molecules, can be effectively utilized in learning given a good filtering strategy. For example, [6] introduces neural fingerprints, which first starts to apply a graph layer on the graph-structure data then followed by deep neural networks.

Such graph-based representations in principle have several advantages. They can contain comprehensive information for molecules, including the skeleton structure, conformational information, and atom features. In contrast, classic molecule representations used in machine learning, such as chemical fingerprints or SMILES, may not be able to encode such information and thus potentially not adequate in some tasks. Furthermore, they are part of deep learning systems, which can be trained end-to-end. On the other hand, the classic molecule representations like chemical fingerprints or SMILES are simple and efficient to calculate, and can be used by different machine learning methods. They can be used by simple methods like random forest or by deep neural networks: a fully-connected neural network on top of fingerprints can lead to a good fit for the data, and both the convolution and recurrent neural networks can map SMILES strings into a latent space for prediction. Furthermore, another disadvantage for graph-based representations is that most efforts focus on message passing between adjacent atoms as representation, which may over-emphasize the local structure and ignore more general information, like the molecule shape. Can we combine the benefits of the two worlds by designing a simple and efficient representation, that can be used by different learning approaches and achieve comparable or even better performance than the existing sophisticated representation methods?

To achieve this, this paper introduces a novel graph-based representation called **N-gram graph**. It first imposes segmented random projection on each atom to get vertex embedding. Then it splits a molecule graph into N-grams with different N's, where an N-gram refers to a path of length N in that graph, and constructs the embedding for each N-gram based on the embeddings of its vertices. The final representation of a molecule is constructed based on the embeddings of all its N-grams. Once constructed, the representations can be fed into different machine learning methods. Furthermore, more localized information is encoded by paths with smaller N, while more globalized information is encoded by paths with larger N. Experiment results support the conclusion that molecule representation has become a bottleneck in virtual screening tasks, and significant gains might be achieved by novel representation methods.

In summary, the contributions of this paper are as follows:

1. We design a novel representation on graph-like data, called **N-gram graph**. It is much simpler than existing graph-based deep neural networks, yet the performance can compete

with the most up-to-date models. And it is very efficient; all the calculations are simple operations like sum and element-wise products.

2. N-gram graph is able to support a finer-grained encoding of the structural information due to the separation of paths of different lengths. This also allows it to balance local and global structural information.

3. The N-gram graph does not require an end-to-end training process, therefore multiple non-deep supervised machine learning methods can be trained on it. Current graph-based deep neural networks apply message passing for information delivery and are only designed for end-to-end deep neural networks, but N-gram graph allows non-deep supervised machine learning methods to reach state-of-the-art performance.

4. The N-gram graph representations show promising generalization performance on deep neural networks. They consistently lead to smaller gaps between training and test performance than existing representations, suggesting that the information is encoded in a way suitable for learning.

Related Work

Deep learning methods started to capture the attention among scientists in the drug discovery domain from Merck Molecular Activity Challange [18, 4]. Efforts expanded to investigate the benefits of multi-task deep neural networks, frequently showing outstanding performance when comparing with shallow models [16, 22, 15]. All of these works used Morgan fingerprints as input representations.

Another option for molecule representation is the SMILES string [24]. SMILES can be treated as a sequence of atoms and bonds, and each molecule has a unique canonical SMILES string among a frequently vast set of noncanonical, but completely valid, SMILES strings. Therefore, attempts were made to make SMILES feed into more complicated neural networks. [11] applied recurrent neural network language model (RNN) and convolutional neural networks (CNN) on SMILES, and showed that CNN is best when evaluated on the log-loss. SMILES as the representation is now common in molecule generation tasks. [9] first applied SMILES for automatic molecule design, and [13] proposed using a parser tree on SMILES so as to produce more grammatically-valid molecules, where the input is the one-hot encoded rules. On the other hand, [15] showed the limitation of SMILES and itself as a structured data is hard to interpret, and thus SMILES are not used in our experiments.

Molecular descriptors [20] is another representation, but it requires heuristically coming up with descriptors and dynamically adjusting it to tasks, which is not easy and requires a lot of domain knowledge. Therefore molecular descriptors are not considered in this paper since one of the goal here is to get a generalized feature representation.

Recent works started to explore the graph representation, and the benefit is its capability to encode the structured data. [6] first utilized message passing on graphs. At each step, this method passes the hidden message layer to the intermediate feature layer. The summed-up neural fingerprints are then fed into neural networks as features. Following this line of research, [1] made small adaptations by using the last message layer as feature inputs for neural network, and [26] proposed a differential pooling layer to learn the hierarchical information.

Other variants introduced different modules. [12] proposed a new module called weave for delivering information among atoms and bonds, and [17] used a weave operation with forward and backward operations across a molecule graph. [14] utilized edge information, and [7] generalized it into a message passing network framework, highlighting the importance of spatial information.

Background and Preliminaries

Generally, molecules can be represented in different formats for machine learning models. The ideal representation should contain comprehensive information for each molecule (like molecule graph) and at the same time easy to learn over for downstream machine learning methods. We consider the following three types of representations in our experiments.

Morgan Fingerprints

Morgan fingerprints and its variants [19] have been one of the most widely used featurization methods in virtual screening. It is an iterative algorithm that encodes the circular substructures of the molecule as identifiers at increasing levels with each iteration. In each iteration, hashing is applied to generate new identifiers, and thus, there is a chance that two substructures are represented by the same identifier. In the end, a list of identifiers encoding the substructures is folded to bit positions of a fixed-length bit string. A 1-bit at a particular position indicates the presence of a substructure (or multiple substructures) and a 0-bit indicates the absence of corresponding substructures.

Graph Representation

Nearly all drug-like molecules can be potentially represented as a graph, where each atom is a vertex and each bond is an edge.

Suppose there are m vertices in the graph, each vertex is denoted by a_i, where $i \in \{0, 1, ...m-1\}$. Each vertex entails useful information, like atom symbol and number of charges for atom nodes. These vertex features are encoded into vertex attribute matrix $\mathcal{N} \in \{0,1\}^{m \times d}$, where d is the dimension of vertex feature. Adjacency matrix $\mathcal{A} \in \{0,1\}^{m \times m}$ is able to depict the skeleton of a graph.

In the molecule graph setting, the attribute vector $\mathcal{N}_{i,\cdot}$ is defined in Equation (1). As in Equation (2), $\mathcal{A}_{i,j} = 1$ if and only if two vertices, a_i and a_j, are linked.

$$\mathcal{N}_{i,\cdot} = [\underbrace{\mathrm{C}, \mathrm{Cl}, \mathrm{I}, \mathrm{F}, \ldots}_{\text{atom symbol}}, \underbrace{0,1,2,3,4,5,6}_{\text{atom degree}}, \ldots, \underbrace{0,1}_{\text{is acceptor}}, \underbrace{0,1}_{\text{is donor}}] \tag{1}$$

$$\mathcal{A}_{i,j} = \begin{cases} 1, & \text{atom}_i \text{ and atom}_j \text{ are bonded} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Each N-gram path in a graph is represented by V, and $|V|$ is the length of that path. A path with length n is represented by V_n. The set of all N-gram paths with same length is called a N-gram path set. These notions will be used in the next section.

Message Passing on Graph Neural Networks

In recent works, message passing has been dominant in graph-based deep neural networks. Message passing has T iterations, corresponding to T layers in deep networks. At step t, each vertex will pass its information only to its neighbors. After continuing for T steps, each vertex is able to pass its own information to vertices at most T-steps away. Therefore, message passing is capable of encoding local structure within the radius of T. The final layer will then aggregate information from all vertices as global representation.

Let the intermediate matrix at step t be $\mathcal{M}_t$, and operation $\mathcal{A} \cdot \mathcal{M}_t$ allows each vertex to pass its own information to its neighbors, where $\cdot$ is the matrix multiplication. Message passing will then multiply it with hidden layer H_t followed by activation function σ. Repeat this process for T times, and the output matrix $\mathcal{M}_T$ is assumed to capture the global information. This process can be nicely written in Equation (3). $\mathcal{M}_0 = \mathcal{N}$ at the initial step.

$$\mathcal{M}_{t+1} = \sigma(H_t[\mathcal{M}_t + \mathcal{A} \cdot \mathcal{M}_t]) \tag{3}$$

One restriction of graph-based representation is that it is only applicable to end-to-end deep neural networks, where all parameters are learned through back-propagation.

N-gram Graph: A Novel Representation

N-gram graph is an order-invariant representation for a graph (e.g., a molecule graph). An N-gram refers to a path of length N in the graph. The method views a given graph as a bag of N-grams and builds representations on them. The high-level process is described as follows:

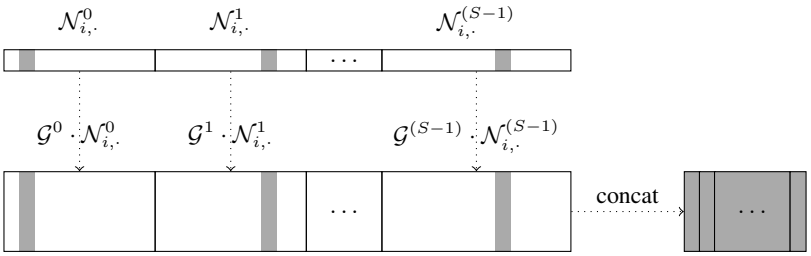


Figure 2: Segmented random projection on vertex a_i. Each vertex's features can be split into S segments. Each group of feature with dimension d_s corresponds to a one-hot vector $N_{i,\cdot}^{s} \in \{0,1\}^{1\times d_s}$ (marked in grey). This vector is then multiplied by the Gaussian random matrix $\mathcal{G}^s \in \mathbb{R}^{r\times d_s}$, yielding a projection into a random space. For each randomized vertex feature g_i, the only non-zero column in output matrix $\mathcal{G}^s \cdot \mathcal{N}_{i,\cdot}^{s}$ in each segment will be extracted and concatenated as the vertex-level embedding.

1. The **vertex** embedding is obtained by applying vertex-level representation called segmented random projection.
2. The **N-gram path** embedding is obtained by the element-wise product of the embeddings of the vertices in the path.
3. For a fixed path length N, sum up all the N-gram paths to get the embedding for the **N-gram path set**.
4. Concatenate the N-gram path sets with multiple N's (e.g., N=1,2,3) to get the final **N-gram graph** representation.

The N-gram graph method is inspired by the N-gram approach in natural language processing (NLP), which is a classical representation method for text data. There, an N-gram refers to N consecutive words in a sentence. A word is mapped to a vector (one-hot vector or other word embeddings). An N-gram can be embedded as the element-wise product of the vectors in it, which corresponds to the diagonal of the tensor product of the word vectors. Summing up all N-gram embeddings and concatenating them for different N lead to the final N-gram embedding for a sentence. This has been shown both theoretically and empirically to preserve good information for downstream learning tasks even using random word vectors (e.g., [23, 2]). We extend this idea from linear graphs (sentences) to general graphs (molecules). We also design our own vertex embedding method to replace the word embedding methods, as detailed below.

The following subsections will go step by step, from problem formulation, to vertex-level representation, and to graph-level representation.

Problem Formulation

For each graph, the goal is to find a set of shared substructures among all the positives. The substructure should preserve related information such as the vertex features and spatial positions. The intuition is that the predicted graphs will have the important properties and be labeled positive if and only if they contain the crucial substructure.

The target is to find such crucial substructure, which can be represented by a candidate set $C \in \{0,1\}^{m\times 1}$, where each bit in C means the corresponding vertex is crucial for the target task. Therefore graph-structured data can be represented in Equation (4).

$$\mathcal{N}^T \cdot C = c_1 \tag{4}$$

$$\mathcal{A} \otimes (C \cdot C^T) \cong c_2 \tag{5}$$

where $\cdot$ is the matrix multiplication, $\otimes$ is the element-wise multiplication, and $A \cong B$ means two matrices, A and B, are isomorphic. For the above constraints in Equation (4), $c_1 \in \mathbb{R}^d$ represents the number of vertices in the crucial substructure, and $c_2 \in \mathbb{R}^m$ is the skeleton for it. Note that here, c_1 and c_2 represent two abstract patterns, and we are not trying to refer them to any specific

components in different algorithms (like feature layers), since they can be quite flexible. To be more specific, if the graph data already contains such patterned structure, then c_1 and c_2 can be directly used as input features; while in the end-to-end deep models, c_1 and c_2 can be learned through back propagation, therefore we treat them as intermediate layers.

One issue for the candidate set C is that it is sensitive to the vertex ordering. Once the indices of vertices are switched, the graph data stays the same but the corresponding representation may change totally. This motivates the order-invariant **N-gram graph** representation.

Vertex Level: Segmented Random Projection

We first apply **segmented random projection** to get the vertex-level embedding. Recall that $\mathcal{A}$ is the adjacency matrix, and $\mathcal{N}$ is the vertex attribute matrix. Vertex features can be treated as S feature segments, where each segment is a one-hot vector. One example is given in Equation (1). Similarly, the vertex attribute matrix can be divided into S segments, $\mathcal{N} = [\mathcal{N}^0, \mathcal{N}^1, \ldots, \mathcal{N}^{(S-1)}]$.

Let $\mathcal{G} = [\mathcal{G}^0, \mathcal{G}^1, \ldots, \mathcal{G}^{(S-1)}] \in \mathbb{R}^{S\times r\times d}$ be a randomized Gaussian matrix, where d is the dimension of the vertex feature and r is the dimension of the random space. It can be divided into S segments according to vertex features. $\mathcal{N}_{i,\cdot}$ is the feature for vertex a_i, and g_i is the corresponding randomized representation. The segmented randomized projection function $f : \mathcal{N}_{i,\cdot} \to g_i$ is defined in Equation (6).

$$\begin{aligned} g_i &= f(\mathcal{N}_{i,\cdot}) \\ &= f([\mathcal{N}^0_{i,\cdot}, \mathcal{N}^1_{i,\cdot}, \ldots, \mathcal{N}^{(S-1)}_{i,\cdot}]) \\ &= [\sum(\mathcal{G}^0 \cdot \mathcal{N}^0_{i,\cdot}), \sum(\mathcal{G}^1 \cdot \mathcal{N}^1_{i,\cdot}), \ldots, \\ &\quad \sum(\mathcal{G}^{(S-1)} \cdot \mathcal{N}^{(S-1)}_{i,\cdot})] \end{aligned} \tag{6}$$

where $\sum$ is the summation along the axis of feature dimension d_s. $\sum(\mathcal{G}^s \cdot \mathcal{N}^s_{i,\cdot}) \in \mathbb{R}^{r\times 1}$ is the random projection on s-th feature segment for a_i. Concatenation of S segments yields $g_i = f(\mathcal{N}_{i,\cdot}) \in \mathbb{R}^{r\times S}$. Figure 2 describes the whole projection process.

Graph Level: N-Gram Graph

Vertex ordering becomes one of the biggest challenges under the current problem formulation. Reordering vertices in one graph will not change its properties, but the candidate set C is not capable of recognizing this difference. Adding an order-invariant representation seems to be a reasonable solution. As mentioned, the N-gram approach is a classic technique used in NLP. It represents a sentence as counts of the contiguous sequence of N words in the sentence. Viewing words as vertices and sentences as linear graphs inspire us to come up with a N-gram method for graph representation.

Each N-gram is a path of length N, and is represented by the element-wise product of the vertices embeddings in that path. Then the embedding for the **N-gram path set** of length n, denoted as $\mathcal{V}_n \in \mathbb{R}^{r\times S}$, $n \in \{1, 2, \ldots, N\}$, is defined as the sum of the embeddings for all n-grams:

$$\mathcal{V}_n = \underbrace{\sum_{\forall V, \text{s.t. } |V|=n} \overbrace{\prod_{a_i \in V} \underbrace{f(\mathcal{N}_{i,\cdot})}_{\text{segmented random projection}}}^{\text{N-gram path}}}_{\text{N-graph path set}} \tag{7}$$

The **N-gram graph** representation is the concatenation of N-gram path sets embeddigns with multiple length n, i.e., $\mathbb{G} = [\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_N] \in \mathbb{R}^{N\times r\times S}$.

Note that N-gram graph representation, each component $\mathcal{V}_n$ of $\mathbb{G}$ corresponds to the path representation with a different length n. Compared to the message-passing in the end-to-end graph-based deep neural networks or Morgan fingerprints generation, the N-gram graph representation can offer a finer-grained view of the graphs, in the sense that it separates different local structures by path

length. Moreover, only path information is involved in the construction of $\mathbb{G}$. But in a message passing graph, important information can get delivered back-and-force along each pair of adjacent vertices, which may lead to a biased representation focusing more on shorter paths.

N-gram graph can be used in different learning methods. It can be flattened into a 1-dimensional vector as input features into non-deep models like random forest or XGBoost. It can also be fed directly into deep neural networks, potentially with flatten operations in the intermediate layers.

Experiments and Results

Recall that machine learning for virtual screening is an efficient way to make drug predictions. Here we test the accuracy of N-gram graph on virtual screening tasks, and compare it with two types of feature representation: Morgan fingerprints and Message-passing graph on end-to-end deep neural networks. N-gram graph proves its effectiveness on various data sets with respect to both the prediction accuracy and generalization ability.

Settings

Six models and three different feature representations were tested on 3 regression tasks and 12 classification tasks. Table 1 lists the feature representation and model combinations. Random Forest (RF) and XGBoost (XGB) [3] are non-deep models. Fully-connected Deep Neural Network (DNN) is deep but not end-to-end. Neural Fingerprints (NEF) [6], Graph CNN (GCNN) [1], and Weave Neural Network (Weave) [12] are end-to-end deep networks and they are only designed for Message-passing graphs.

Table 1: Feature representation for each different machine learning model. Here we have 9 different combinations. Both Morgan fingerprints and N-gram graph are trained on non-deep models and fully-connected deep neural networks.

Model	Feature Representation
NEF	Message-passing graph
GCNN	Message-passing graph
Weave	Message-passing graph
RF	Morgan fingerprints / N-gram graph
XGB	Morgan fingerprints / N-gram graph
DNN	Morgan fingerprints / N-gram graph

Table S2 lists the specific $d = 42$ features and $S = 8$ segments for the vertices. All data sets are split into five folds with one selected as hold-out test set. We follow the hyperparameters provided in [6, 1, 12] for NEF, GCNN and Weave respectively. For other models, we run a comprehensive grid search for hyperparameter sweeping, including two non-deep machine learning algorithms, RF and XGB. More details about hyperparameters are provided in the appendix. Furthermore, for models using N-gram graph, the effects of the random projection dimension r and the N-gram dimension N will be discussed in the appendix, while the following sections display results with $N = 6$ and $r = 100$. (The effect of N and r are discussed in the appendix). All the codes will be public on GitHub.

Regression Tasks

Table 2: RMSE on three regression tasks (test set). Top three results after 5-fold cross-validation are **bolded**, and standard deviation values are included in Table S9. Baseline results (*) are from [6, 12].

Representation	Morgan			Message-passing graph			N-gram graph		
Method	RF	XGB	DNN	NEF(*)	GCNN	Weave(*)	RF	XGB	DNN
delaney	1.311	1.110	1.231	**0.520**	0.913	**0.460**	0.773	0.700	**0.699**
malaria	**1.028**	**1.008**	1.052	1.160	1.055	1.070	1.030	**1.010**	1.119
cep	1.642	1.410	1.477	1.430	**1.184**	**1.100**	1.379	**1.290**	1.365

Compound representations were compared on three different regression tasks from the same data sets used in the previous work [6].

- **Delaney:** 1144 molecules were measured with respect to the aqueous solubility [5].
- **Malaria:** [8] measures the drug efficacy of 10,000 molecules against the parasite that causes malaria.
- **CEP:** A subset of 20,000 molecules from Havard Clean Energy Project (CEP) [10]. It aims at estimating organic photovoltaic efficiency.

As demonstrated in Table 2, performance on regression tasks varies a lot. When comparing N-gram graph with Morgan fingerprints, all three models can obtain better RMSE. Message-passing graph shows slightly better performance on Delaney and CEP, but other models based on N-gram graph are very comparative.

Classification Tasks

Table 3: AUC[ROC] on test set on Tox21. Top three results after 5-fold cross-validation are **bolded**, and standard deviations are included in Table S6. Each row corresponds to a task, except that last row measures the general performance over all tasks.

Featurization	Morgan			Message-passing graph			N-gram graph		
Method	RF	XGB	DNN	NEF	GCNN	Weave	RF	XGB	DNN
NR-AR	0.787	0.777	0.756	0.723	0.793	**0.796**	**0.802**	0.790	**0.795**
NR-AR-LBD	**0.864**	0.852	0.817	0.813	**0.858**	0.816	0.844	**0.858**	0.853
NR-AhR	**0.903**	**0.900**	0.854	0.841	0.896	0.869	0.890	**0.898**	0.869
NR-Aromatase	0.827	0.802	0.742	0.738	0.824	**0.830**	**0.845**	**0.852**	0.830
NR-ER	0.724	0.721	0.692	0.673	**0.734**	**0.729**	0.727	**0.733**	0.712
NR-ER-LBD	**0.815**	0.783	0.772	0.725	0.805	0.804	**0.810**	**0.819**	0.787
NR-PPAR-gamma	**0.839**	0.793	0.756	0.758	**0.821**	0.803	0.801	**0.825**	0.783
SR-ARE	**0.818**	**0.809**	0.781	0.740	0.782	0.790	0.808	**0.826**	0.777
SR-ATAD5	**0.857**	0.828	0.738	0.763	**0.839**	0.823	**0.841**	0.837	0.811
SR-HSE	**0.793**	0.764	0.731	0.702	**0.774**	0.771	0.773	**0.786**	0.750
SR-MMP	0.886	0.879	0.856	0.856	**0.888**	0.886	**0.895**	**0.909**	0.865
SR-p53	**0.849**	0.823	0.759	0.782	**0.840**	0.813	0.833	**0.843**	0.805
average	**0.830**	0.811	0.771	0.760	0.821	0.811	**0.822**	**0.831**	0.803

Table 4: Generalization performance: Train and test gap on AUC[ROC]. Top three results after 5-fold cross-validation are **bolded**, and standard deviations are included in Table S7.

Representation	Morgan			Message-passing graph			N-gram graph		
Task	RF	XGB	DNN	NEF	GCNN	Weave	RF	XGB	DNN
NR-AR	0.213	0.209	0.243	0.277	**0.125**	**0.115**	0.198	0.210	**0.093**
NR-AR-LBD	**0.136**	0.144	0.180	0.187	**0.115**	0.156	0.156	0.142	**0.112**
NR-AhR	0.097	0.091	0.146	0.154	**0.054**	**0.059**	0.110	0.102	**0.048**
NR-Aromatase	0.172	0.190	0.258	0.262	**0.112**	**0.099**	0.155	0.148	**0.066**
NR-ER	0.274	0.238	0.307	0.319	**0.129**	**0.129**	0.273	0.267	**0.032**
NR-ER-LBD	0.184	0.205	0.228	0.274	**0.134**	**0.119**	0.189	0.181	**0.105**
NR-PPAR-gamma	0.161	0.199	0.244	0.241	**0.142**	**0.147**	0.197	0.175	**0.147**
SR-ARE	0.181	0.166	0.219	0.255	**0.118**	**0.099**	0.192	0.174	**0.075**
SR-ATAD5	0.143	0.167	0.262	0.234	**0.125**	**0.129**	0.159	0.163	**0.123**
SR-HSE	0.206	0.222	0.269	0.296	**0.155**	**0.155**	0.225	0.214	**0.095**
SR-MMP	0.114	0.109	0.144	0.144	**0.069**	**0.063**	0.105	0.091	**0.047**
SR-p53	0.151	0.170	0.241	0.215	**0.107**	**0.112**	0.167	0.157	**0.064**
average	0.169	0.176	0.229	0.238	**0.115**	**0.115**	0.177	0.169	**0.084**

Tox21: "Toxicology in the 21st Century" [21] initiative created a public database measuring toxicity of compounds, which was used in the 2014 Tox21 Data Challenge. Table S1 shows the sizes of the tasks. Nine representation and model pairs are tested on twelve tasks, and Table 3 summarizes the AUC[ROC] on the test set. Overall, we observe that N-gram graph indeed leads to comparable or even better performance than the other approaches. Details are discussed below.

After a thorough hyperparameter sweeping, both RF on Morgan fingerprints and N-gram graph and XGB on N-gram graph in Table 3 prove to be top three algorithms. Other algorithms, like GCNN and

Weave, also exhibit competitive performance. Switching from Morgan fingerprints to N-gram graph appears to benefit XGB and DNN performance, especially for DNN: we observe the performance on 11 out of 12 tasks get improved, and 7 out of 12 taks get improved on XGB. Such huge improvements show the effectiveness of N-gram graph.

Random forest on Morgan fingerprints has performance beyond general expectation. One possible explanation is that we have used 4000 trees and obtained improved performance compared to 75 trees as in [25]. This is not surprising, since the number of trees is the most important parameter as pointed out in [15]. Another possible reason is that Morgan fingerprints indeed contains sufficient amount of information for the classification tasks, and methods like random forest are good at exploiting them while deep neural networks are not.

If only deep neural networks are considered, GCNN and Weave are generally better than DNN and NEF in Tox21 data set. This reveals that N-gram graph may be a better fit to decision tree models, yet DNN shows most robust performance as will be discussed below. Future work could be done to investigate more suitable deep neural network structures.

Generalization Performance on Tox21

An advantage of the N-gram graph representations is that they lead to good generalization performance, as shown in Table 4. In particular, when used by deep neural networks, they consistently lead to the smallest gaps between train and test performance. This demonstrates that they encode the information needed for classification in a way suitable for the neural networks to extract. It is also observed that the Morgan fingerprints typically has large gaps (statistical analysis in Figure S3). This is probably because that the information encoded by the hashing scheme is hard to exploit by the learning methods. This then requires larger hypothesis class to fit the feature, resulting in larger gaps. In contrast, a small gap for N-gram graph suggests that the information is encoded in a learning friendly way, easy for learning methods to exploit.

Conclusion

This paper introduces a novel representation method called **N-gram graph** for graph structured data, and applies it on molecule property predictions in the virtual screening tasks. The first step in tackling this task is the derivation of problem formulation which requires an order-invariant representation. Then, the idea of N-gram from NLP is taken and extended to graph data. The resulting representation can be used by most supervised machine learning methods. Comprehensive experiments show the potential benefits of N-gram graph by reaching state-of-the-art performance on many benchmarks using different classification and regression models.

This work provides a simple, principled, and effective method to handle the graph-structured data. In areas like image classification, representations learned by the convolution layer and other techniques have allowed deep learning methods to take a dominant role. We believe in the domains with graph-structured data like molecules, similar great improvements would be accessible with both novel representation and novel algorithm. Concrete future work can be generalizing the word embedding methods and attention mechanisms from NLP.

Acknowledgements

This work was supported in part by FA9550-18-1-0166. The authors would also like to acknowledge computing resources from the University of Wisconsin-Madison Center for High Throughput Computing and support provided by the University of Wisconsin-Madison Office of the Vice Chancellor for Research and Graduate Education with funding from the Wisconsin Alumni Research Foundation.

References

[1] Han Altae-Tran, Bharath Ramsundar, Aneesh S Pappu, and Vijay Pande. Low data drug discovery with one-shot learning. *ACS Central Science*, 3(4):283–293, 2017.

[2] Sanjeev Arora, Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. A compressed sensing view of unsupervised text embeddings, bag-of-n-grams, and lstm. *International Conference on Learning Representations*, 2018.

[3] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2016.

[4] George Dahl. Deep learning how i did it: Merck 1st place interview. *Online article available from http://blog. kaggle. com/2012/11/01/deep-learning-how-i-did-it-merck-1st-place-interview*, 2012.

[5] John S. Delaney. ESOL: Estimating Aqueous Solubility Directly from Molecular Structure. *Journal of Chemical Information and Computer Sciences*, 44(3):1000–1005, May 2004.

[6] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alan Aspuru-Guzik, and Ryan P Adams. Convolutional Networks on Graphs for Learning Molecular Fingerprints. pages 2224–2232, 2015.

[7] Felix A Faber, Luke Hutchison, Bing Huang, Justin Gilmer, Samuel S Schoenholz, George E Dahl, Oriol Vinyals, Steven Kearnes, Patrick F Riley, and O Anatole von Lilienfeld. Prediction errors of molecular machine learning models lower than hybrid dft error. *Journal of chemical theory and computation*, 13(11):5255–5264, 2017.

[8] Francisco-Javier Gamo, Laura M. Sanz, Jaume Vidal, Cristina de Cozar, Emilio Alvarez, Jose-Luis Lavandera, Dana E. Vanderwall, Darren V. S. Green, Vinod Kumar, Samiul Hasan, James R. Brown, Catherine E. Peishoff, Lon R. Cardon, and Jose F. Garcia-Bustos. Thousands of chemical starting points for antimalarial lead identification. *Nature*, 465(7296):305–310, May 2010.

[9] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 2016.

[10] Johannes Hachmann, Roberto Olivares-Amaya, Sule Atahan-Evrenk, Carlos Amador-Bedolla, Roel S. Sánchez-Carrera, Aryeh Gold-Parker, Leslie Vogt, Anna M. Brockway, and Alán Aspuru-Guzik. The Harvard Clean Energy Project: Large-Scale Computational Screening and Design of Organic Photovoltaics on the World Community Grid. *The Journal of Physical Chemistry Letters*, 2(17):2241–2251, September 2011.

[11] Stanisław Jastrzębski, Damian Leśniak, and Wojciech Marian Czarnecki. Learning to smile (s). *arXiv preprint arXiv:1602.06289*, 2016.

[12] 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.

[13] Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. *arXiv preprint arXiv:1703.01925*, 2017.

[14] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

[15] Shengchao Liu, Moayad Alnammi, Spencer S Ericksen, Andrew F Voter, James L Keck, F Michael Hoffmann, Scott A Wildman, and Anthony Gitter. Practical model selection for prospective virtual screening. *bioRxiv*, page 337956, 2018.

[16] Junshui Ma, Robert P Sheridan, Andy Liaw, George E Dahl, and Vladimir Svetnik. Deep neural nets as a method for quantitative structure–activity relationships. *Journal of chemical information and modeling*, 55(2):263–274, 2015.

[17] Matthew K. Matlock, Na Le Dang, and S. Joshua Swamidass. Learning a Local-Variable Model of Aromatic and Conjugated Systems. *ACS Central Science*, 4(1):52–62, January 2018.

[18] Merck. Merck molecular activity challenge. *https://www.kaggle.com/c/MerckActivity*, 2012.

[19] HL Morgan. The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. *Journal of Chemical Documentation*, 5(2):107–113, 1965.

[20] Roberto Todeschini and Viviana Consonni. *Molecular descriptors for chemoinformatics: volume I: alphabetical listing/volume II: appendices, references*, volume 41. John Wiley & Sons, 2009.

[21] Tox21 Data Challenge. Tox21 data challenge 2014. *https://tripod.nih.gov/tox21/challenge/*, 2014.

[22] Thomas Unterthiner, Andreas Mayr, Günter Klambauer, Marvin Steijaert, Jörg K Wegner, Hugo Ceulemans, and Sepp Hochreiter. Deep learning as an opportunity in virtual screening. *Advances in neural information processing systems*, 27, 2014.

[23] Sida Wang and Christopher D Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 90–94. Association for Computational Linguistics, 2012.

[24] David Weininger, Arthur Weininger, and Joseph L Weininger. Smiles. 2. algorithm for generation of unique smiles notation. *Journal of Chemical Information and Computer Sciences*, 29(2):97–101, 1989.

[25] 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.

[26] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L Hamilton, and Jure Leskovec. Hierarchical graph representation learning withdifferentiable pooling. *arXiv preprint arXiv:1806.08804*, 2018.

Task Specification

Table S1: Number of positives and total number of molecules in 12 Tox21 tasks.

Task	Num of Positives	Total Number
NR-AR	304	7332
NR-AR-LBD	237	6817
NR-AhR	783	6592
NR-Aromatase	298	5853
NR-ER	784	6237
NR-ER-LBD	347	7014
NR-PPAR-gamma	186	6505
SR-ARE	954	5907
SR-ATAD5	262	7140
SR-HSE	378	6562
SR-MMP	912	5834
SR-p53	414	6814

Atom Feature Specification

Table S2 shows the types of features and feature segments for the atoms in the molecules of the data sets used in our experiments.

Table S2: $d = 42$ features are divided into $S = 8$ segments. Each segment of features correspond to one type of atom property, including atom symbol, atom degree, atom charge, etc. Note that the number of atom symbols can be pretty large, so we use the last bit 'Unknown' as the placeholder to catch the missing symbols.

id	digit	property	values
0	0-9	atom symbol	[C, Cl, I, F, O, N, P, S, Br, Unknown]
1	10-16	atom degree	[0, 1, 2, 3, 4, 5, 6]
2	17-23	number of Hytrogeon	[0, 1, 2, 3, 4, 5, 6]
3	24-29	implicit valence	[0, 1, 2, 3, 4, 5]
4	30-35	atom charge	[-2, -1, 0, 1, 2, 3]
5	36-37	is aromatic	[no, yes]
6	38-39	is acceptor	[no, yes]
7	40-41	is donor	[no, yes]

Hyperparameter Search

For Neural Fingerprints, Graph CNN and Weave Neural Network, we follow the hyperparameters provided in [6, 1, 12] respectively. For other models, we run a comprehensive grid search for hyperparameter sweeping, including random forest in Table S3, XGBoost in Table S4, and fully-connected deep neural network in Table S5.

Table S3: Hyperparameter sweeping for random forest.

Hyperparameters	Candidate values
n_estimators	4000, 8000, 16000
max_features	None, sqrt, log2
min_samples_leaf	1, 10, 100, 1000
class_weight	None, balanced_subsample, balanced

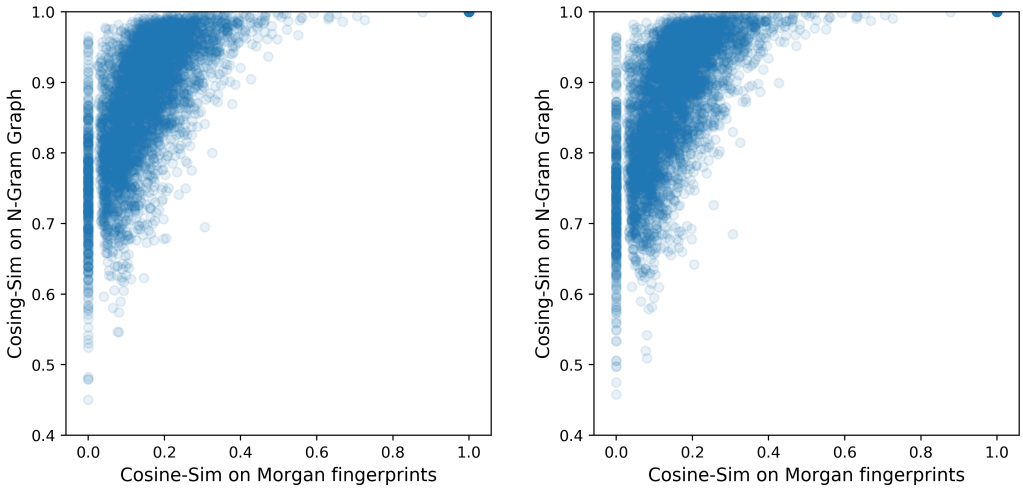
Table S4: Hyperparameter sweeping for XGBoost.

Hyperparameters	Candidate values
max_depth	5, 10, 50, 100
learning_rate	1, 3e-1, 1e-1, 3e-2
n_estimators	30, 100, 300, 1000, 3000

Table S5: Hyperparameter sweeping for fully-connected deep neural network.

Hyperparameters	Candidate values
batch_size	128, 256, 512
epoch	100, 500, 1000
network structure	[50, 30], [150, 50]

Qualitative Analysis



(a) $N = 6, r = 50$ (b) $N = 6, r = 100$

Figure S1: Comparison of pairwise molecule similarities between Morgan fingerprints and N-gram graph. Each point corresponds to one molecule point where y-axis is the cosine similarity on N-gram graph and x-axis is the cosine similarity on Morgan fingerprints. Larger random dimension ($r = 100$ on the right) shows slightly wider distribution than lower dimension ($r = 50$ on the left). The vertical band of points on the left side of the plots show compounds with no detectable similarities (cosine similarity=0) by Morgan fingerprints that do show various levels of similarities when comparing N-gram graph.

To further explain how N-gram graph can help with representation, pairwise cosine similarities on different representations are compared. Here we randomly select 100 molecules from Delaney data set [5].

As displayed in Figure S1, similarities based on Morgan fingerprints tend to concentrate around 0, while N-gram graph is inclined to make molecules concentrate on similar representation. Some molecules are observed to be overlapped on Morgan fingerprints while N-gram graph is able to distinguish among them.

Statistical Analysis

Test Performance on Tox21 Tasks

The complete 5-fold cross validation results are displayed in Table S6 and we run Tukey's test as shown in Figure S2.

Table S6: AUC[ROC] on test set on Tox21. Top three results after 5-fold cross-validation are **bolded**. Each row corresponds to a task, except that last row measures the general performance over all tasks. N-gram graph representation uses $N = 6$ and $r = 100$. Results on other values of N and r are displayed in Table S10.

Featurization	Morgan			Message-passing graph			N-gram graph		
Method	RF	XGB	DNN	NEF	GCNN	Weave	RF	XGB	DNN
NR-AR	0.787 ±0.07	0.777 ±0.06	0.756 ±0.06	0.723 ±0.04	0.793 ±0.07	**0.796 ±0.06**	**0.802 ±0.08**	0.790 ±0.07	**0.795 ±0.06**
NR-AR-LBD	**0.864 ±0.06**	0.852 ±0.05	0.817 ±0.06	0.813 ±0.07	**0.858 ±0.04**	0.816 ±0.05	0.844 ±0.04	**0.858 ±0.03**	0.853 ±0.03
NR-AhR	**0.903 ±0.03**	**0.900 ±0.02**	0.854 ±0.04	0.841 ±0.05	0.896 ±0.02	0.869 ±0.04	0.890 ±0.02	**0.898 ±0.02**	0.869 ±0.02
NR-Aromatase	0.827 ±0.07	0.802 ±0.06	0.742 ±0.10	0.738 ±0.06	0.824 ±0.05	**0.830 ±0.05**	**0.845 ±0.07**	**0.852 ±0.05**	0.830 ±0.06
NR-ER	0.724 ±0.02	0.721 ±0.02	0.692 ±0.02	0.673 ±0.04	**0.734 ±0.04**	**0.729 ±0.02**	0.727 ±0.04	**0.733 ±0.04**	0.712 ±0.02
NR-ER-LBD	**0.815 ±0.05**	0.783 ±0.06	0.772 ±0.02	0.725 ±0.08	0.805 ±0.02	0.804 ±0.03	**0.810 ±0.06**	**0.819 ±0.04**	0.787 ±0.04
NR-PPAR-gamma	**0.839 ±0.04**	0.793 ±0.09	0.756 ±0.04	0.758 ±0.08	**0.821 ±0.11**	0.803 ±0.06	0.801 ±0.10	**0.825 ±0.10**	0.783 ±0.11
SR-ARE	**0.818 ±0.04**	**0.809 ±0.04**	0.781 ±0.05	0.740 ±0.03	0.782 ±0.04	0.790 ±0.05	0.808 ±0.03	**0.826 ±0.02**	0.777 ±0.05
SR-ATAD5	**0.857 ±0.05**	0.828 ±0.07	0.738 ±0.08	0.763 ±0.09	**0.839 ±0.04**	0.823 ±0.04	**0.841 ±0.03**	0.837 ±0.04	0.811 ±0.02
SR-HSE	**0.793 ±0.03**	0.764 ±0.04	0.731 ±0.03	0.702 ±0.04	**0.774 ±0.04**	0.771 ±0.04	0.773 ±0.05	**0.786 ±0.07**	0.750 ±0.06
SR-MMP	0.886 ±0.02	0.879 ±0.03	0.856 ±0.03	0.856 ±0.03	**0.888 ±0.02**	0.886 ±0.02	**0.895 ±0.02**	**0.909 ±0.02**	0.865 ±0.02
SR-p53	**0.849 ±0.03**	0.823 ±0.06	0.759 ±0.03	0.782 ±0.09	**0.840 ±0.05**	0.813 ±0.07	0.833 ±0.03	**0.843 ±0.05**	0.805 ±0.04
average	**0.830 ±0.05**	0.811 ±0.05	0.771 ±0.05	0.760 ±0.05	0.821 ±0.05	0.811 ±0.04	**0.822 ±0.04**	**0.831 ±0.05**	0.803 ±0.04

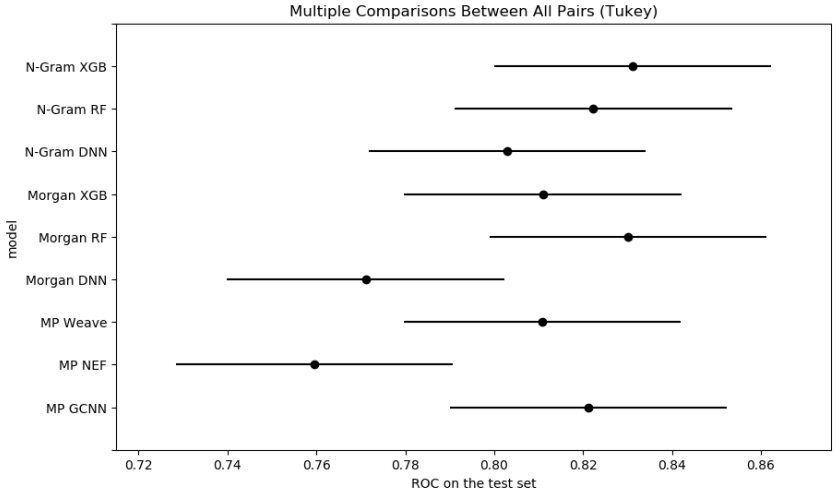


Figure S2: Tukey's test on test performance (AUC[ROC]).

Generalization Performance on Tox21 Tasks

The generalization performance for complete 5-fold cross validation is displayed in Table S7 and we run Tukey's test as shown in Figure S3.

Table S7: Generalization performance: Train and test gap on AUC[ROC]. Top three results after 5-fold cross-validation are **bolded**. N-gram graph representation uses $N = 6$ and $r = 100$. Though its performance has not reached the best of all, N-gram graph with DNN is the most robust pair.

Representation	Morgan			Message-passing graph			N-gram graph		
Task	RF	XGB	DNN	NEF	GCNN	Weave	RF	XGB	DNN
NR-AR	0.213 ±0.07	0.209 ±0.06	0.243 ±0.06	0.277 ±0.04	**0.125 ±0.08**	**0.115 ±0.08**	0.198 ±0.08	0.210 ±0.07	**0.093 ±0.07**
NR-AR-LBD	**0.136 ±0.06**	0.144 ±0.05	0.180 ±0.05	0.187 ±0.07	**0.115 ±0.04**	0.156 ±0.04	0.156 ±0.04	0.142 ±0.03	**0.112 ±0.05**
NR-AhR	0.097 ±0.03	0.091 ±0.02	0.146 ±0.04	0.154 ±0.05	**0.054 ±0.02**	**0.059 ±0.05**	0.110 ±0.02	0.102 ±0.02	**0.048 ±0.03**
NR-Aromatase	0.172 ±0.07	0.190 ±0.06	0.258 ±0.10	0.262 ±0.06	**0.112 ±0.06**	**0.099 ±0.06**	0.155 ±0.07	0.148 ±0.05	**0.066 ±0.08**
NR-ER	0.274 ±0.02	0.238 ±0.02	0.307 ±0.02	0.319 ±0.04	**0.129 ±0.03**	**0.129 ±0.04**	0.273 ±0.04	0.267 ±0.04	**0.032 ±0.03**
NR-ER-LBD	0.184 ±0.05	0.205 ±0.06	0.228 ±0.02	0.274 ±0.08	**0.134 ±0.03**	**0.119 ±0.06**	0.189 ±0.06	0.181 ±0.04	**0.105 ±0.07**
NR-PPAR-gamma	0.161 ±0.04	0.199 ±0.09	0.244 ±0.04	0.241 ±0.08	**0.142 ±0.11**	**0.147 ±0.06**	0.197 ±0.11	0.175 ±0.10	**0.147 ±0.13**
SR-ARE	0.181 ±0.04	0.166 ±0.04	0.219 ±0.05	0.255 ±0.03	**0.118 ±0.05**	**0.099 ±0.06**	0.192 ±0.03	0.174 ±0.02	**0.075 ±0.08**
SR-ATAD5	0.143 ±0.05	0.167 ±0.07	0.262 ±0.08	0.234 ±0.09	**0.125 ±0.04**	**0.129 ±0.04**	0.159 ±0.03	0.163 ±0.04	**0.123 ±0.03**
SR-HSE	0.206 ±0.03	0.222 ±0.05	0.269 ±0.03	0.296 ±0.04	**0.155 ±0.05**	**0.155 ±0.04**	0.225 ±0.05	0.214 ±0.07	**0.095 ±0.07**
SR-MMP	0.114 ±0.02	0.109 ±0.03	0.144 ±0.03	0.144 ±0.03	**0.069 ±0.03**	**0.063 ±0.02**	0.105 ±0.02	0.091 ±0.02	**0.047 ±0.04**
SR-p53	0.151 ±0.03	0.170 ±0.06	0.241 ±0.03	0.215 ±0.10	**0.107 ±0.06**	**0.112 ±0.07**	0.167 ±0.03	0.157 ±0.05	**0.064 ±0.05**
average	0.169 ±0.05	0.176 ±0.04	0.229 ±0.05	0.238 ±0.05	**0.115 ±0.03**	**0.115 ±0.03**	0.177 ±0.04	0.169 ±0.05	**0.084 ±0.03**

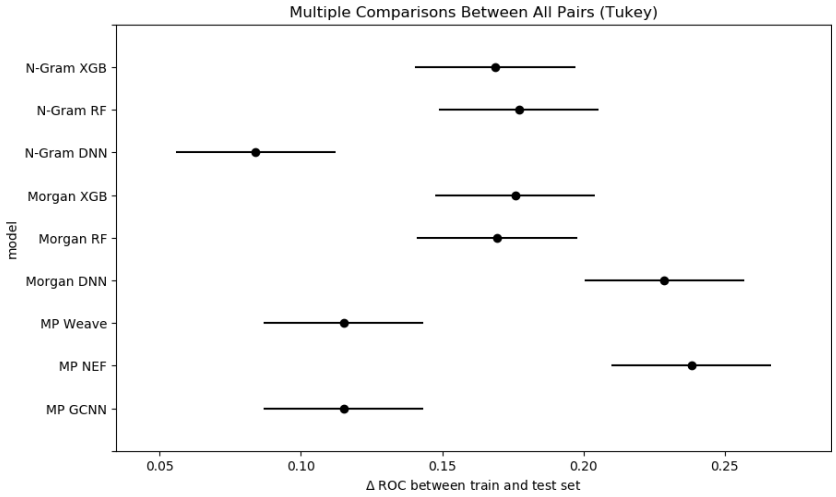


Figure S3: Tukey's test on generalization performance (AUC[ROC]).

Test on Models with Different Vector Embeddings

Table S8: For each message-passing graph method, compare the performance on 12 Tox21 tasks.

Group 1	Group 2	mean diff	reject
NEF new embedding	NEF original embedding	0.0004	False
GCNN new embedding	GCNN original embedding	-0.0012	False
Weave new embedding	Weave original embedding	0.0008	False

To further prove that different vector embeddings are not biasing the message-passing graph methods, we test the original embeddings proposed in the previous papers [6, 1, 12] and the ones in Table S2. The null hypothesis here is that means are the same, so rejection=False means we should accept the null hypothesis. Thus Table S8 shows that two vector embeddings contain very similar information.

Test Performance on Delaney, Malaria, and CEP

Table S9: RMSE on three regression tasks (test set). Top three results are **bolded** and the best performance is **underlined**. Baseline results (*) are from [6, 12]. $r = 100$ and $N = 6$ in N-gram graph. Results with other r and N are displayed in Table S11, Table S12, and Table S13.

Representation	Morgan			Message-passing graph			N-gram graph		
Method	RF	XGB	DNN	NEF(*)	GCNN	Weave(*)	RF	XGB	DNN
delaney	1.311 ±0.17	1.110 ±0.13	1.231 ±0.11	**0.520 ± 0.14**	0.913 ±0.06	**0.460 ± 0.16**	0.773 ±0.08	0.700 ±0.09	**0.699 ± 0.05**
malaria	**1.028 ± 0.03**	**1.008 ± 0.03**	1.052 ±0.05	1.160 ±0.06	1.055 ±0.04	1.070 ±0.12	1.030 ±0.02	**1.010 ± 0.03**	1.119 ±0.03
cep	1.642 ±0.03	1.410 ±0.04	1.477 ±0.04	1.430 ±0.18	**1.184 ± 0.06**	**1.100 ± 0.12**	1.379 ±0.01	**1.290 ± 0.03**	1.365 ±0.03

Results on Classification Tasks (Tox21)

We run N-gram graph on 12 classification tasks from "Toxicology in the 21st Century" [21]. We tested the effects of random projection dimension r and N-gram dimension N, and the ROC on validation sets are listed in Table S10.

Table S10: This table includes three different methods on N-gram graph. Two values for r and three valeus for N are tested. For each combination, model with best performance is **bolded**.

target name	r	N	XGBoost	RF	DNN
NR-AR	50	2	0.825	0.825	**0.855**
		4	0.832	0.826	**0.863**
		6	0.839	0.826	**0.864**
	100	2	0.818	**0.826**	0.816
		4	0.832	0.823	**0.847**
		6	**0.837**	0.822	0.830
NR-AR-LBD	50	2	0.835	0.851	**0.867**
		4	0.835	0.849	**0.857**
		6	0.845	0.841	**0.857**
	100	2	0.835	0.855	**0.867**
		4	0.827	0.841	**0.857**
		6	0.843	0.840	**0.845**
NR-AhR	50	2	**0.883**	0.874	0.852
		4	**0.889**	0.872	0.852
		6	**0.884**	0.870	0.851
	100	2	**0.887**	0.874	0.866
		4	**0.888**	0.873	0.861
		6	**0.886**	0.872	0.859
NR-Aromatase	50	2	0.839	**0.849**	0.826
		4	0.829	**0.849**	0.824
		6	0.833	**0.849**	0.826
	100	2	0.829	**0.853**	0.824
		4	0.833	**0.848**	0.834
		6	0.829	**0.844**	0.832
NR-ER	50	2	**0.712**	0.693	0.708
		4	**0.717**	0.695	0.708
		6	0.704	0.697	**0.708**
	100	2	0.711	0.694	**0.717**
		4	0.714	0.698	**0.719**
		6	0.704	0.699	**0.729**
NR-ER-LBD	50	2	**0.811**	0.801	0.805
		4	**0.821**	0.816	0.800
		6	**0.829**	0.812	0.799
	100	2	**0.822**	0.798	0.813
		4	**0.821**	0.818	0.801
		6	**0.822**	0.807	0.802
NR-PPAR-gamma	50	2	0.821	**0.850**	0.726
		4	0.784	**0.847**	0.728
		6	0.817	**0.826**	0.717
	100	2	0.802	**0.849**	0.751
		4	0.802	**0.835**	0.748
		6	0.792	**0.837**	0.772
SR-ARE	50	2	**0.819**	0.815	0.795
		4	**0.829**	0.824	0.807
		6	0.826	**0.827**	0.804
	100	2	**0.822**	0.815	0.799
		4	**0.837**	0.828	0.803
		6	**0.836**	0.832	0.794

SR-ATAD5	50	2	**0.853**	0.840	0.814
		4	0.846	**0.865**	0.807
		6	0.844	**0.858**	0.807
	100	2	**0.858**	0.844	0.805
		4	0.853	**0.865**	0.821
		6	0.843	**0.858**	0.808
SR-HSE	50	2	**0.785**	0.760	0.775
		4	**0.805**	0.771	0.779
		6	**0.821**	0.773	0.771
	100	2	**0.792**	0.762	0.759
		4	**0.798**	0.775	0.760
		6	**0.796**	0.771	0.764
SR-MMP	50	2	**0.897**	0.887	0.857
		4	**0.904**	0.893	0.851
		6	**0.905**	0.893	0.849
	100	2	**0.903**	0.889	0.863
		4	**0.909**	0.893	0.860
		6	**0.908**	0.893	0.862
SR-p53	50	2	**0.847**	0.826	0.778
		4	**0.864**	0.840	0.778
		6	**0.872**	0.843	0.772
	100	2	**0.855**	0.830	0.791
		4	**0.868**	0.841	0.795
		6	**0.865**	0.841	0.794
Average	50	2	**0.827**	0.823	0.805
		4	**0.830**	0.829	0.804
		6	**0.835**	0.826	0.802
	100	2	**0.828**	0.824	0.806
		4	**0.832**	0.828	0.809
		6	**0.830**	0.826	0.807

Result On Regression Tasks (Delaney, Malaria, CEP)

We run N-gram graph on 3 regression tasks, Delaney, Malaria, and CEP. We tested the effects of random projection dimension r and N-gram dimension N, and the RMSE on validation sets are listed in Table S11, Table S12, and Table S13 respectively.

Table S11: Three models with different combinations of r and n. Evaluated on task Delaney.

target name	r	n	XGBoost	RF	DNN
Delaney	50	1	0.826	0.805	**0.707**
		2	0.772	0.802	**0.678**
		4	**0.771**	0.807	0.837
		6	0.780	0.819	**0.666**
	100	1	0.804	0.804	**0.670**
		2	0.782	0.800	**0.745**
		4	0.806	0.809	**0.686**
		6	0.783	0.820	**0.713**

Table S12: Three models with different combinations of r and n. Evaluated on task Malaria.

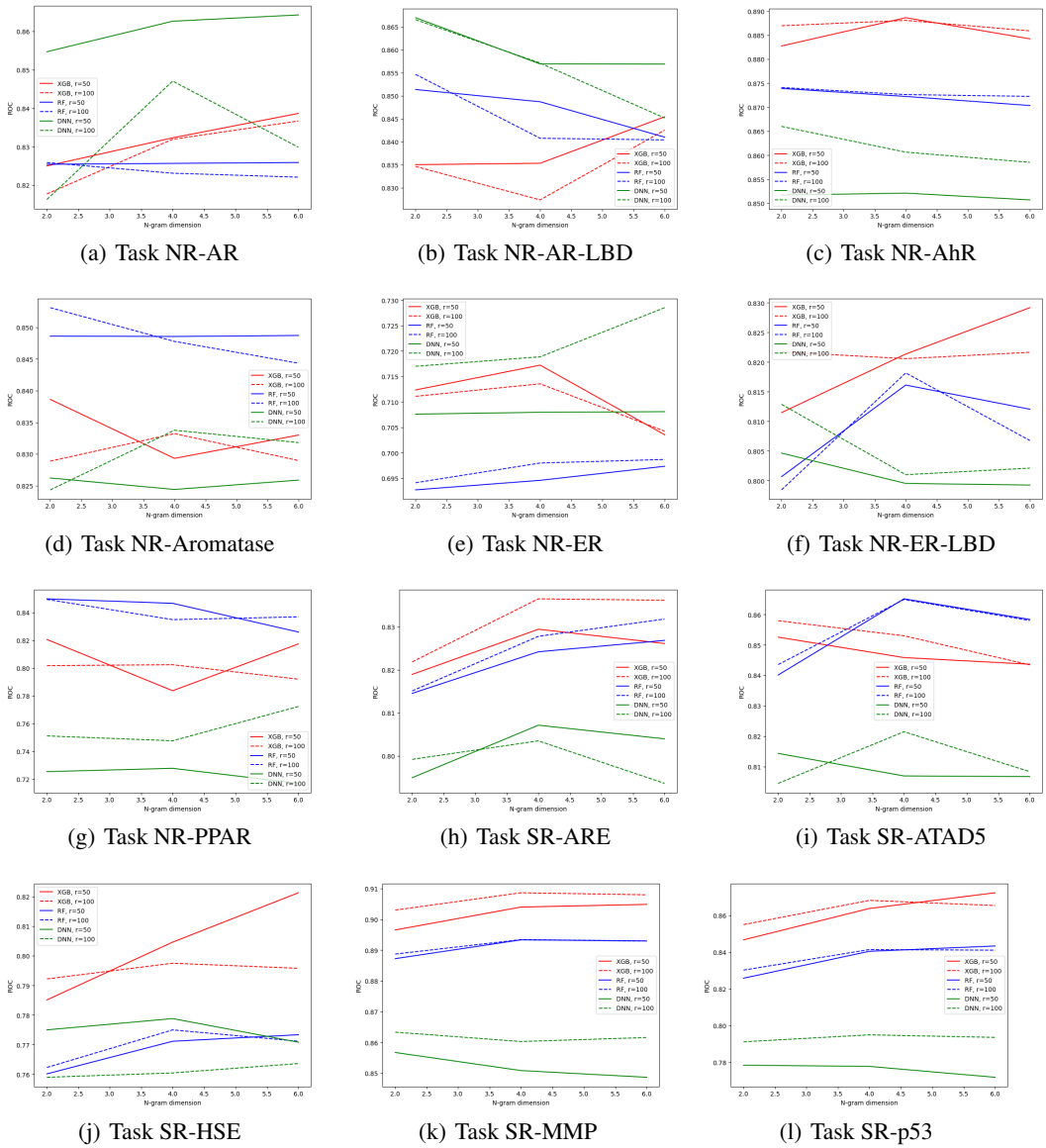
target name	r	n	XGBoost	RF	DNN
Malaria	50	1	1.079	**1.059**	1.112
		2	**1.033**	1.038	1.109
		4	**1.006**	1.016	1.102
		6	**1.003**	1.013	1.085
	100	1	1.072	**1.054**	1.145
		2	1.036	**1.034**	1.129
		4	**1.007**	1.012	1.133
		6	**0.991**	1.011	1.106

Table S13: Three models with different combinations of r and n. Evaluated on task CEP.

target name	r	n	XGBoost	RF	DNN
CEP	50	1	1.645	1.644	**1.540**
		2	1.487	1.496	**1.409**
		4	**1.322**	1.377	1.383
		6	**1.288**	1.374	1.359
	100	1	1.646	1.642	**1.582**
		2	1.472	1.490	**1.414**
		4	**1.330**	1.372	1.357
		6	**1.296**	1.367	1.344

Effects of r and N

As observed from Figure S4, for the 12 tasks from Tox21, the ROC values on the validation set are not converging as r and N increases. Two possible reasons for this: (1) Data is insufficient. As shown in Table S1, all tasks have less than 8000 molecules. (2) ROC reveals the ranking of predictions, while some other metrics, like RMSE shown in Figure S5, are more likely to depict the predictions in a finer-grained way.



(a) Task NR-AR (b) Task NR-AR-LBD (c) Task NR-AhR

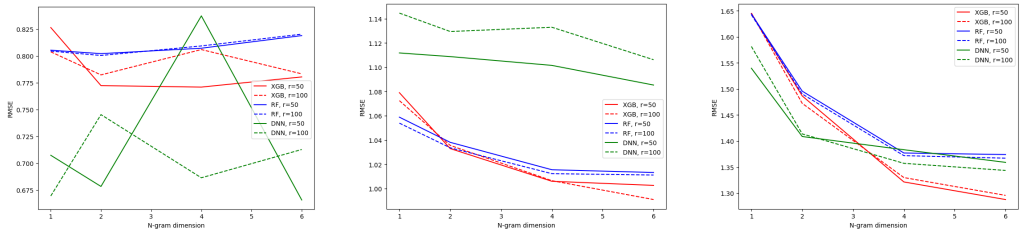
(d) Task NR-Aromatase (e) Task NR-ER (f) Task NR-ER-LBD

(g) Task NR-PPAR (h) Task SR-ARE (i) Task SR-ATAD5

(j) Task SR-HSE (k) Task SR-MMP (l) Task SR-p53

Figure S4: Effects of random projection dimension r and N-gram dimension N on 12 tasks from Tox21: how the ROC on validation set changes as different r and N.

As observed from Figure S5, for task Malaria and CEP, increasing N can help reduce the loss, while different values of random projection dimension r show very similar performance. Performance on Delaney Figure S5 fluctuates a lot as r and N increases. One conjecture is that such high variance might be caused by the data insufficiency (only 1144 molecules are contained in this dataset). However, we can still conclude that for each machine learning algorithm, $r = 100$ and $N = 6$ are reasonable to choose.



(a) Task Delaney (b) Task Malaria (c) Task CEP

Figure S5: Effects of random projection dimension r and N-gram dimension N on tasks Delaney, Malaria and CEP: how the RMSE on validation set changes as different r and N.