JCIM JOURNAL OF CHEMICAL INFORMATION AND MODELING

pubs.acs.org/jcim Article

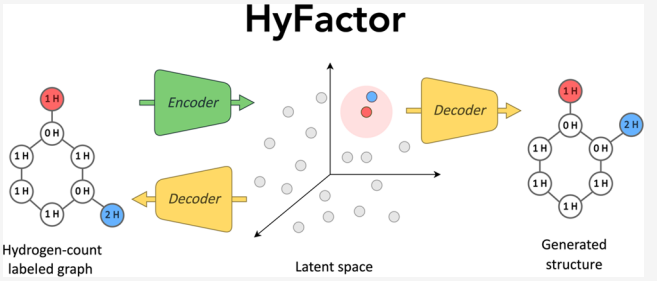
HyFactor: A Novel Open-Source, Graph-Based Architecture for Chemical Structure Generation

Tagir Akhmetshin, Arkadii Lin, Daniyar Mazitov, Yuliana Zabolotna, Evgenii Ziaikin, Timur Madzhidov,* and Alexandre Varnek*

Cite This: *J. Chem. Inf. Model.* 2022, 62, 3524–3534 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: Graph-based architectures are becoming increasingly popular as a tool for structure generation. Here, we introduce novel open-source architecture HyFactor in which, similar to the InChI linear notation, the number of hydrogens attached to the heavy atoms was considered instead of the bond types. HyFactor was benchmarked on the ZINC 250K, MOSES, and ChEMBL data sets against conventional graph-based architecture ReFactor, representing our implementation of the reported DEFactor architecture in the literature. On average, HyFactor models contain some 20% less fitting parameters than those of ReFactor. The two architectures display similar validity, uniqueness, and reconstruction rates. Compared to the training set compounds, HyFactor generates more similar structures than ReFactor. This could be explained by the fact that the latter generates many open-chain analogues of cyclic structures in the training set. It has been demonstrated that the reconstruction error of heavy molecules can be significantly reduced using the data augmentation technique. The codes of HyFactor and ReFactor as well as all models obtained in this study are publicly available from our GitHub repository: https://github.com/Laboratoire-de-Chemoinformatique/HyFactor.



INTRODUCTION

Nowadays, deep neural networks (DNNs) play a significant role in drug and material discovery, being used for property prediction,[1] de novo design,[2] and computer-aided retrosynthesis.[3] One of the most widely used DNN architectures is the autoencoder (AE).[4] It is able not only to encode chemical structures in their latent representation but also to generate new compounds by decoding sampled latent vectors using a decoder subnetwork.

To generate new molecular structures, the majority of AEs use SMILES strings[5] as an input, which allows one to employ the power of natural language processing (NLP) techniques. Although SMILES seems suitable for de novo design tasks, the latent representation of text strings may not reflect chemical similarity relationships between considered structures.

Graph-based AE (GAE) architectures[6] serve as a valuable alternative to the SMILES-based ones. They present a chemical structure as a graph in which nodes and edges encode atoms and chemical bonds, respectively. GAEs have three fundamental advantages over SMILES-based autoencoders. First, no specific order of graph traversal is required, which solves the problem of fixing the canonical ordering of atoms or training on random ordering. Second, a graph object does not need to follow specific grammar rules, such as opening and closing brackets, cycle numbering, etc., which seriously limits the generation ability of neural networks. Notice that different non-canonical SMILESs describing the same structure may be embedded to different latent vectors. Finally, GAEs always generate graph objects, which, in turn, allows for a meaningful chemical analysis of errors including detection of graph disconnectivity and erroneous valence.

A molecular graph can be represented by an ensemble of atom vectors and bond matrices, which, in turn, can be transformed into its vector representation using graph convolutional networks (GCN).[7] Once a latent vector is obtained, it can be decoded using either single-shot or iterative decoders. Single-shot decoders generate atoms and bonds in a graph in a single pass.[6,8] Their training is fast, but simultaneous generation of atom vectors and bond matrices is technically challenging.[9] In contrast, iterative decoders create atoms and bonds sequentially one by one until a molecule is reconstructed.[10] Iterative decoders can be categorized into two classes. The first class generates atom vectors one by one until vectors of all atoms are sampled. These vectors are then used to extract atom and bond types. For example, in the

Received: June 12, 2022
Published: July 25, 2022

https://doi.org/10.1021/acs.jcim.2c00744
J. Chem. Inf. Model. 2022, 62, 3524–3534

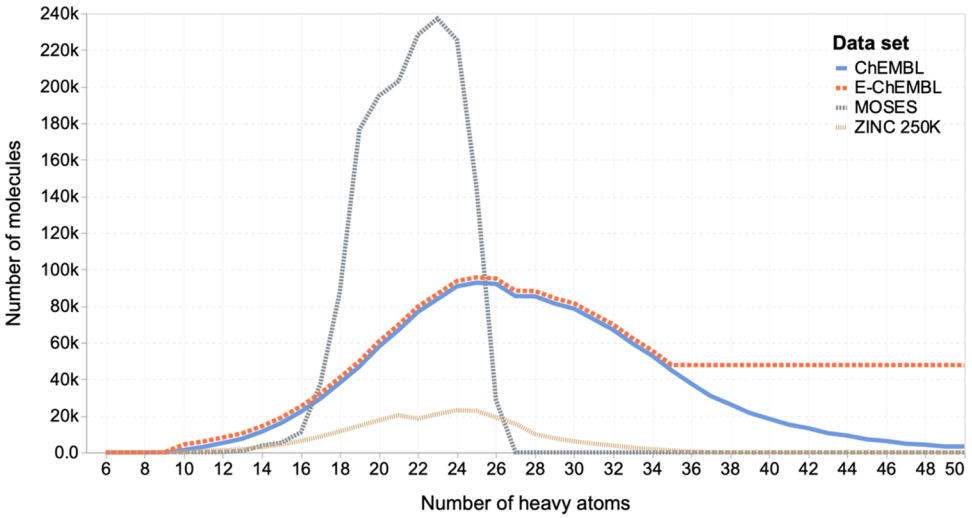


Figure 1. Heavy-atom count distribution in studied data sets.

autoregressive method, the generation of the next vector is based on previously created atom vectors.[11] Another popular approach employs a recurrence-based generation where the next atom vector is generated from a hidden (or difference) vector updated at every step.[12] The second class of decoders uses a Markov decision process. In contrast with previous methods, they require the explicit creation of the molecule's substructure at each step of generation. To achieve that, they perform several actions such as "creation of atoms" and "creation of bonds" until the molecule is generated.[10] However, iterative generation requires a much more complex and slow network architecture compared to single-shot autoencoders.[8] One of the most efficient recurrence-based iterative decoder architectures was recently implemented in the DEFactor tool reported by Assouel et al. in the arXiv e-print.[11] The encoder in DEFactor is a multi-layer GCN, whereas the decoder combines the long short-term memory (LSTM)[13] cell for atomic vector generation with a new adjacency matrix defactorization procedure.

Typically, the GCN employed by the encoder subnetwork in GAEs computes the neighbors' messages within each bond-type specific channel. For this reason, it is necessary to store up to four bond-type-specific adjacency matrices and specific trainable weight matrices. This takes a lot of memory and requires numerous mathematical operations with the corresponding computational graph. A complex iterative process of atom and bond "creation" and related high memory and time costs prevent GAE architectures from becoming widely used.

In this paper, we propose an alternative to conventional structure encoding, which may help reduce both the required GPU memory and the model training time. Instead of considering different bond types, we propose to use the number of hydrogens attached to each heavy atom, similar to the InChI linear notation.[14] Together with an adjacency matrix, this information is sufficient to reproduce molecular structures. Also, it solves the problem of a standard representation of functional groups and aromaticity. In this case, a molecular graph can be represented by three objects: (1) a vector of atoms, (2) a vector of hydrogen counts, and (3) a binary adjacency matrix. The above approach was implemented in a new hydrogen-count labeled graph-based defactorization (HyFactor) GAE architecture. In HyFactor, a DNN is combined with the algorithm needed to convert a regular molecular graph to a hydrogen-count labeled graph (a graph where a certain number of hydrogens is assigned to each heavy atom) and back.

In order to assess the efficiency of the new architecture compared to conventional GAE, we have decided to compare HyFactor with DEFactor. However, neither DEFactor codes nor neural network hyperparameters (e.g., the number of convolutional layers in the encoder or the batch size, etc.) needed for re-implementation of this tool were provided in the original publication.[11] Therefore, we attempted to re-implement and further improve the DEFactor architecture in its advanced version referred here as ReFactor. Here, we describe the HyFactor and ReFactor architectures and report the benchmarking results on ZINC250K, ChEMBL v. 27, and MOSES data sets in the reconstruction and generation tasks.

METHODS

Data and Curation. Three data sources were used: ZINC250K benchmarking data set extracted from the ZINC database[4] by Kusner et al.,[15] MOSES data set from the MOSES package (v. 1.0),[16] and ChEMBL database (v. 27).[17] All sets were standardized with ChemAxon JChem's utilities[18] using the following procedures: (1) dearomatization, (2) isotope removal, (3) stereo mark removal, (4) explicit hydrogen removal, (5) small fragment removal, (6) solvent removal, (7) salt strip, (8) neutralization of charges, (9) functional group transformation, (10) selection of the canonical tautomer form of the molecule, (11) aromatization, (12) duplicate removal, and (13) dearomatization. The order of atoms in standardized structures was defined by the canonical SMILES string produced by ChemAxon JChem.

Some 1.7K structures were removed from the ZINC250K data set as a result of the cleaning procedure. The cleaned set was split into training, validation (tuning), and test sets as reported by Kusner et al.[15] The validation set was used for early stopping.[19] The test set consisted of 5K predefined molecules, and the remaining structures were randomly split into training and validation sets in a ratio of 9:1. Note that several duplicates were found in ZINC250K (see Table S1) due to the presence of stereoisomers in the data set. This may cause some overestimation of the performance of the earlier reported models.

https://doi.org/10.1021/acs.jcim.2c00744
J. Chem. Inf. Model. 2022, 62, 3524−3534

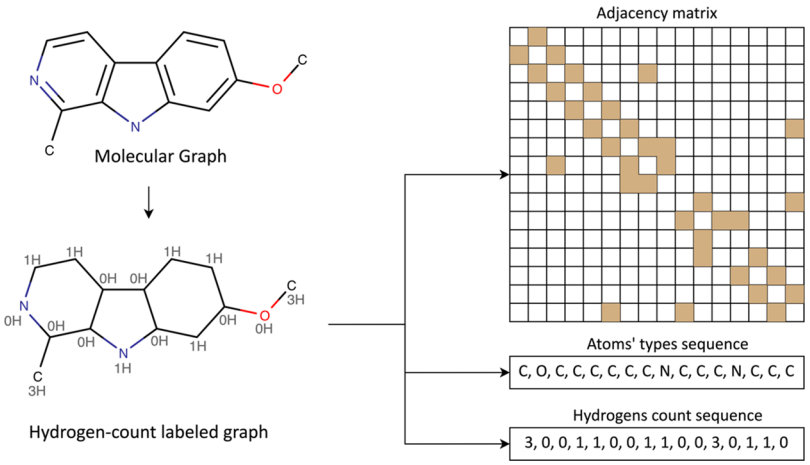


Figure 2. Hydrogen-count labeled graph representation. Here, the molecular graph (hydrogens are hidden) is converted into a graph with no edge features, while the nodes have two features, namely, the type of atoms and number of hydrogens.

The MOSES data set is a benchmarking set for generative models (details are given in the Supporting Information). It was analyzed with the proposed standardization procedure; however, no mistakes were found. Therefore, it was used as it is. The original "training" set was split into training and validation sets in a 4:1 ratio.

The ChEMBL database was standardized by the same workflow as the ZINC250K data set. The initial database consisted of 1.9M molecules, and it was reduced to 1.6M of standardized structures. The prepared data set was additionally analyzed in terms of the frequency of atom types (Figure S1). Sixty unique atomic types (including information on atomic charges) were found in ChEMBL, and molecules containing only 15 atomic types (C, O, N, S, F, Cl, N^+, O^-, Br, P, I, N^-, B, Si, and Se) have been retained according to the threshold of 1000. The compounds containing less than 5 and more than 50 heavy atoms have been discarded due to their under-representation. The filtrated ChEMBL data set was then split into a training set (80% of data or 1.3M molecules) and a test set (20% of data or 327K molecules). We did not use a validation set for early stopping since, starting from 100 epochs, the model performance parameters (the loss and reconstruction rate) practically did not vary. The modeling was stopped at 150 epochs.

All chemical structures from each data set have been kekulized in order to avoid a need to introduce an additional aromatic bond type for the ReFactor architecture. The latter would increase the size of the graph-based architecture, slow down the calculations, and decrease the number of valid structures in sampling.

Both the ChEMBL and ZINC250K data sets have a similar distribution of heavy atom counts (Figure 1). However, the MOSES data set differs from both, and most of the structures lie in the range from 16 to 26 heavy atoms. For some computational tests, the ChEMBL database was enriched by 460K virtual structures bearing more than 35 heavy atoms. The heavy-atom distribution of the enriched ChEMBL (E_ChEMBL) is shown in Figure 1.

Hydrogen-Count Labeled Graph. In the hydrogen-count labeled graph (HLG), only the connections between atoms and the count of hydrogens connected to the atoms are taken into account (see Figure 2). The formal charge of an atom is used as a vertex label. Such a representation has already been tested in the development of structure−property models with graph convolution networks.[20] The implemented workflow first transformed a molecular graph to HLG and then to three complementary representations: adjacency matrix, atomic types, which include the atom symbol and charge, and hydrogen-count vectors (Figure 2). The conversion from a molecular graph to HLG and back was performed with the help of the CGRtools toolkit.[21]

Autoencoders Based on Conventional (DEFactor and ReFactor) and Hydrogen-Count (HyFactor) Representations of the Molecular Graph. All three autoencoder architectures used in this work, namely, DEFactor (reported earlier)[11] and ReFactor and HyFactor (both developed in this work), are depicted in Figure 3. Their detailed description is given below.

DEFactor Architecture. The encoder in DEFactor uses one-hot embedding to represent atoms in the molecular graph and consists of several layers of edge-specific graph convolution networks[7,22] that can be expressed as

$$\mathbf{H}^{l+1} = \mathrm{ReLU}\left[\sum_b (\mathbf{D}_b^{-1/2}\mathbf{E}_b\mathbf{D}_b^{-1/2}\mathbf{H}^l\mathbf{W}_b^l) + \mathbf{H}^l\mathbf{W}_{\mathrm{self}}^l\right] \quad (1)$$

where $\mathbf{H}^l$ is the atoms' vectors after the lth graph convolution layer, $\mathbf{E}_b$ is a bond-type specific adjacency matrix, $\mathbf{D}_b$ is the corresponding bond-type specific diagonal degree matrix, $\mathbf{W}_b$ and $\mathbf{W}_{\mathrm{self}}$ are trainable matrices of weights for every bond type b and weights for self-channel, respectively, and ReLU is the rectifier activation function. The aggregation of atomic vectors is performed with the help of a long short-term memory (LSTM)[13] unit followed by a one-layer perceptron (see Figure 3a), giving a molecular latent vector.

In the decoder, the molecular latent vector is unpacked into a set of atomic embeddings, $\tilde{\mathbf{H}}$, where each h_i is predicted using the LSTM layer. Thus, the entire matrix of atoms' embedding is restored. Next, it passes through two subunits in parallel where the first one is represented by a multilayer perceptron (MLP) with the *softmax* activation function that returns predictions of atom types ($\tilde{A}$). The second subunit realizes the multichannel defactorization procedure[23] needed to reconstruct the bond matrix according to eq 2

$$\tilde{\mathbf{E}}_b = \sigma(\tilde{\mathbf{H}}\mathbf{W}_b\tilde{\mathbf{H}}^{\mathrm{T}} + \mathrm{bias}) \quad (2)$$

https://doi.org/10.1021/acs.jcim.2c00744
J. Chem. Inf. Model. 2022, 62, 3524−3534

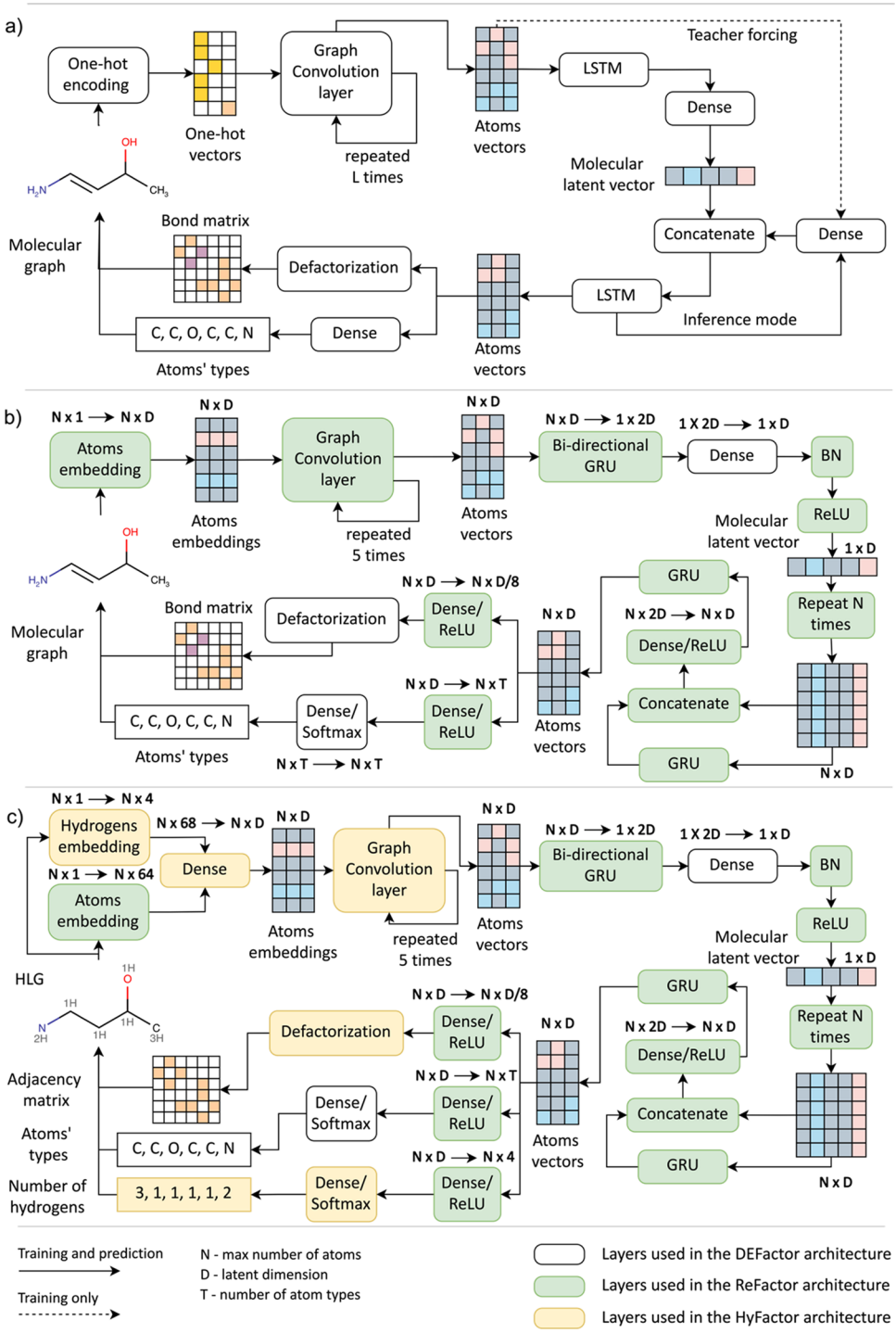


Figure 3. Architectures of different autoencoders considered in this work: (a) DEFactor,[11] (b) ReFactor, and (c) HyFactor. BN refers to the batch normalization layer, and GRU refers to the gated recurrent unit. Parameters for each experiment are specified in Table S2.

where $\tilde{\mathbf{E}}_b$ is the reconstructed adjacency matrix for a bond type b, $\tilde{\mathbf{H}}$ is the matrix of the recovered atoms' embeddings h_i, $\mathbf{W}_b$ is a diagonal matrix of weights for the bond type b, and σ is the sigmoid activation function. A certain probability is returned for each bond type between each pair of atoms, and the bond type with the highest probability is selected for the reconstruction. A three-step procedure including teacher forcing was used to speed up the DEFactor training. Within each step, trainable weights of a certain part of the network were frozen and then relaxed at the next step.

The loss function is a sum of categorical cross-entropy for atom predictions (eq 3) and binary cross-entropy for bond predictions (eq 4)

$$L_{\text{atoms}} = -\frac{1}{n}\sum A \times \log(\tilde{A}) \tag{3}$$

$$L_{\text{bonds}} = -\frac{1}{n^2}\sum_{b}^{4}[E_b \times \log(\tilde{E}_b) + (1 - E_b) \times \log(\tilde{E}_b)] \tag{4}$$

https://doi.org/10.1021/acs.jcim.2c00744
J. Chem. Inf. Model. 2022, 62, 3524−3534

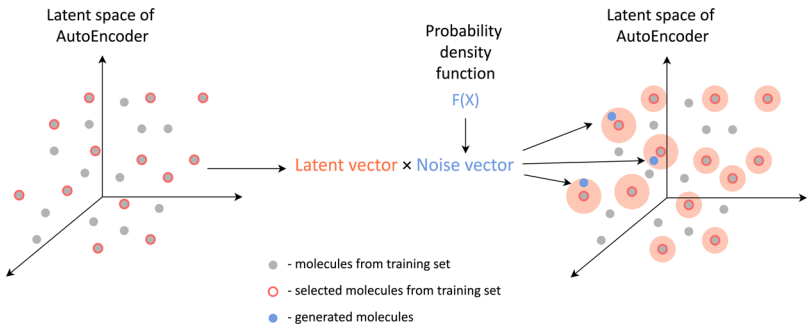


Figure 4. Sampling of new structures from the autoencoder latent space.

where A is the one-hot matrix for atom types, $\tilde{A}$ is the predicted atom-type probability matrix, n is the number of atoms in a molecule, and E_b is the real adjacency matrix for bond type b.

It should be noted that some important information like the number of GCN layers as well as the dimensionality of the atoms' embedding matrix was missed in the original publication by Assouel et al.[11] Therefore, we reimplemented the DEFactor architecture with some modifications that improved its performance, at least, for large molecules (see below).

ReFactor Architecture. The ReFactor architecture keeps the main ideas of the DEFactor model. Some parts of DEFactor that were explained well were kept and reimplemented in the Tensorflow package[24] v. 2.6. Others were modified or replaced by new layers. Thus, it was decided that token embedding is a more powerful and flexible technique than a simple one-hot embedding. Hence, the latter was replaced by a token embedding layer.

To stabilize the learning process, the GCN from the DEFactor was completed by a layer normalization (LN)[25] layer among the atoms' features (parameter "axis = −2") and masking of imaginary atoms (padding):

$$\mathbf{H}^{l+1} = \text{mask} \times \text{ReLU}\left[\sum_b (\mathbf{D}_b^{-1/2}\mathbf{E}_b\mathbf{D}_b^{-1/2} \times \text{LN}(\mathbf{H}^l\mathbf{W}_b^l)) + \text{LN}(\mathbf{H}^l\mathbf{W}_{\text{self}}^l)\right] \quad (5)$$

In each experiment, the number of GCN layers was fixed at five. The dimensionality of the input and output vectors did not change across the layers.

For atomic vector aggregation, LSTM units in DEFactor were replaced with bidirectional gated recurrent units[26] (GRU; see Figure 3b). The output of GRUs was passed to the dense, batch normalization (BN), and ReLU activation layers to obtain the molecular latent vector. A teacher-forcing technique applied in the original article was skipped since no predictive performance improvement was detected in our experiments.

In the decoder, the atoms' vectors were generated by two sequentially connected GRU layers and a dense layer in between headed by a RepeatVector layer (see Figure 3b). In such a case, the input molecular latent vector was repeated N times (i.e., according to the maximal molecular graph size) and passed through the first GRU, and intermediate atom vectors were returned. These intermediate vectors were then concatenated with the repeated molecular vectors. They passed first through a perceptron layer with a ReLU activation function and then through the second GRU layer. Further, the hidden vectors of the second GRU were used as the retrieved atoms' embeddings. For the atom reconstruction, the retrieved atoms' embeddings were passed through two dense layers with the output dimensionality equal to the number of atom types. Activation of the first layer was ReLU, and activation of the second was the *softmax* function. During the bond reconstruction step, the atoms' embeddings were forwarded to a dense layer with the output dimensionality of the latent vector divided by 8 and ReLU activation and then to the defactorization layer.

These and other minor changes allowed us to handle molecules containing up to 50 heavy atoms (see Results and Discussion). Unless specified, the dimensionality of the atom embedding vectors as well as all internal and latent vectors was the same. Parameters for each experiment are specified in Table S2. All layers were taken with standard parameters if not specially mentioned.

HyFactor Architecture. The main changes compared to ReFactor concern the steps of graph convolution and graph reconstruction from the atoms' vectors (see Figure 3c). First, the HLG was transformed to the atoms' (dimension of 64) and hydrogens' (dimension of 4) embeddings. These embeddings were concatenated and passed through the dense layer with the ReLU activation function. The graph convolution network was similar to that in ReFactor

$$\mathbf{H}^{l+1} = \text{mask} \times \text{ReLU}[\mathbf{D}^{-1/2}\mathbf{E}\mathbf{D}^{-1/2} \times \text{LN}(\mathbf{H}^l\mathbf{W}_{\text{neighbors}}^l) + \text{LN}(\mathbf{H}^l\mathbf{W}_{\text{self}}^l)] \quad (6)$$

where $\mathbf{E}$ and $\mathbf{D}$ are adjacency and degree matrices of HLG, and the other designations are the same as in eqs 1 and 2. Here, the number of the training parameters is twice less than that for ReFactor GCN.

At the graph reconstruction stage, the number of hydrogens and atom types were predicted similar to using dense layers with the *softmax* activation function. The maximal number of hydrogens attached to an atom was equal to 3. The adjacency matrix was reconstructed using the defactorization procedure (eq 2), ignoring bond types; only one trainable diagonal $\mathbf{W}_b$ matrix was used.

The initialization of weights for each layer was performed with HE normal initialization.[27] Training of the architecture was performed using the AdaBelief optimizer[28] with default parameters. Exponential learning decay was applied in order to maintain the training stability.

Sampling Procedure. New molecular structures were generated by sampling latent vectors in the vicinity of the known molecules.[29] Here, the latent vectors of selected

https://doi.org/10.1021/acs.jcim.2c00744
J. Chem. Inf. Model. 2022, 62, 3524−3534

molecules from the training set were used as seeds (Figure 4). During generation, these latent vectors were multiplied by noise vectors composed of random numbers generated from a probability density function of a log-normal distribution followed by their decoding to a molecular graph.

In order to effectively explore the chemical space around the given seed, the "mean" parameter was set to 0, whereas the "standard deviation" was systematically varied. Generally, the probability of generating more dissimilar structures increases with the "standard deviation" value.

RESULTS AND DISCUSSION

Reconstruction Rate of Different Graph Autoencoders. The performance of the autoencoders' model is measured by the reconstruction rate representing a percentage of correctly reconstructed structures in the considered data set. Reconstruction rate values for several SMILES-based and graph-based autoencoders on the ZINC 250K set are reported in Table 1.

Table 1. ZINC 250K Reconstruction Benchmarking Results[a]

architecture name	molecular representation	reconstruction rate (%)		
		training set	validation set	test set
TSGCD[9]	molecular graph			90.5
DEFactor[11]	molecular graph			89.8
JTVAE[12]	molecular graph[b]			76.7
rebalanced VAE[30]	SMILES			92.7
all SMILES[31]	SMILES			87.6
SDVAE[32]	SMILES			76.2
ReFactor	molecular graph	99.5	90.8	90.7
ReFactor[c]	molecular graph	99.7	90.0	90.0
HyFactor	HLG	99.3	89.3	89.0
HyFactor[c]	HLG	99.2	89.8	88.4

[a]Reconstruction rate values for other architectures are taken from the original publications. [b]JTVAE uses hierarchical fragments instead of atoms to reconstruct the molecule. [c]Additional standardization and duplicate data removal have been applied to the initial ZINC 250K data set.[15]

One can see that the performances of graph-based and SMILES-based architectures are similar. The leading graph-based architecture (TSGCD) has a reconstruction rate that is only 2% lower than the best SMILES-based autoencoder (rebalanced VAE). The DEFactor architecture also demonstrates high performance, which is only 1% lower than the leading graph-based autoencoder. Notice that the data standardization issue was not sufficiently discussed in the publications on all the architectures mentioned in Table 2. Therefore, results for the ReFactor and HyFactor architectures are given for both initial[15] (non-standardized) and standardized data sets. In addition, results for training and validation sets for model overfitting analysis are also reported.

The reconstruction rate for ReFactor obtained on the initial data set is slightly better than that of the standardized data set. This fact supports our assumption that the performance of autoencoder may be overestimated since training and test sets partially overlap in the non-standardized data. As it follows from Table 1, ReFactor outperforms DEFactor on the initial data set.

Table 2. MOSES Metrics[a] Calculated for the 100 K Structures Generated with Different Standard Deviations (STDs)[d]

STD	validity		uniqueness		novelty	
	original[b]	modified[c]	original[b]	modified[c]	original[b]	modified[c]
ReFactor						
0.2	0.997	0.996	0.656	0.105	0.074	0.042
0.4	*0.921*	*0.886*	*0.748*	*0.265*	*0.634*	*0.578*
0.6	0.698	0.503	0.948	0.730	0.875	0.814
0.8	0.540	0.149	0.998	0.971	0.978	0.855
1	0.516	0.022	0.961	1.000	0.999	0.983
HyFactor						
0.2	1.000	0.9880	0.661	0.193	0.117	0.006
0.4	*0.989*	*0.7290*	*0.795*	*0.267*	*0.729*	*0.129*
0.6	0.940	0.1820	0.980	0.634	0.923	0.288
0.8	0.886	0.0120	1.000	0.966	0.993	0.491
1	0.822	0.0003	0.981	1.000	1.000	0.912

[a]Validity is a fraction of valid molecules compared to generated ones. Uniqueness (or Unique 10K) is defined as a fraction of the first 10K unique molecules among the valid ones. Novelty is a fraction of the novel generated molecules among unique ones. [b]Metrics calculated by the MOSES package. [c]Metrics calculate by CGRtools after removal of structures with valence errors and disconnected graphs. [d]Results for STD = 0.4 selected for further tests are shown in italics.

The HyFactor architecture has almost the same reconstruction rate as ReFactor but has a smaller number of neural network parameters (10M compared to 12M in ReFactor). The training time for HyFactor was 2 h and 45 min. (with 5 GB of GPU RAM allocated), while for ReFactor, it was 4 h and 10 min. (with 5.5 GB of GPU RAM allocated).

Structure Generation Performance of Graph-Based Autoencoders. The efficiency of the proposed graph-based autoencoders to generate valid chemical structures has been investigated on the MOSES data set using the metrics included in the MOSES package.[16] Both ReFactor and HyFactor architectures were trained on 80% of the MOSES training set and achieved more than 99% of the reconstruction rate on the remaining 20% of the data used as a validation set. Then, 10 K compounds were randomly selected as seeds for sampling new structures. For each seed compound, 10 virtual structures were generated, so 100K structures were obtained. The generation was based on a log-normal distribution with a mean equal to 0and a standard deviation ranging from 0.2 to 1.0 with a step of 0.2. Since the original DEFactor source code was not available from the original publication,[11] the ReFactor architecture was benchmarked instead.

A preliminary analysis of the results revealed that the MOSES package did not correctly handle typical problems frequently occurring during the structure generation: disconnected molecular graphs were not considered erroneous, and some valence errors were ignored. Therefore, an additional examination of generated structures was performed using the CGRtools package.[21] The results of the analysis of STD influence on validity, uniqueness, and novelty metrics of the generated structures are given in Table 2.

One can see that the increase in standard deviation leads, on one hand, to the rise of the percentage of new molecules and to the decrease of validity on the other hand. We notice the difference between original and modified workflows for validity checks caused mainly by the trend to return disconnected graphs by ReFactor and HyFactor at high STD, which

https://doi.org/10.1021/acs.jcim.2c00744
J. Chem. Inf. Model. 2022, 62, 3524−3534

Table 3. Results of MOSES Benchmarking for Different Autoencoders; the Similarity Metrics FCD, SNN, and Scaf[a] Relate a Set of Generated Structures with the MOSES Test Set[d]

model	validity	uniqueness	novelty	FCD	SNN	Scaf	IntDiv
AAE	0.937	0.997	0.793	0.556	0.608	0.902	0.856
VAE	0.977	0.998	0.695	0.099	0.626	0.939	0.856
JTVAE	1.000	1.000	0.914	0.395	0.548	0.896	0.855
ReFactor[b]	0.886	0.265[c]	0.578	1.743	0.547	0.847	0.868
HyFactor[b]	0.729	0.267[c]	0.129	0.365	0.614	0.862	0.860

[a]The MOSES benchmarking parameters:[16] Scaf is a cosine similarity based on the occurrence of Bemis−Murcko scaffolds in the compared sets. SNN is an average Tanimoto similarity calculated with Morgan fingerprints between a molecule from the generated set and its nearest neighbor from the test set. FCD is a Wasserstein-2 distance computed on vectors produced by the last layer of the ChemNet neural network between the generated and test sets. IntDiv measures the dissimilarity of structures in the generated set calculated with Morgan fingerprints. See the Supporting Information for details. [b]Sampling for STD = 0.4. All metrics were calculated after the removal of structures with valence errors and disconnected structures using the CGRtools-based workflow. [c]Uniqueness was calculated on the entire generated data set after filtration by validity. [d]The performances of AAE, VAE (SMILES-based), and JTVAE (graph-based) architectures were taken from the article by Polykovskiy et al.[16]

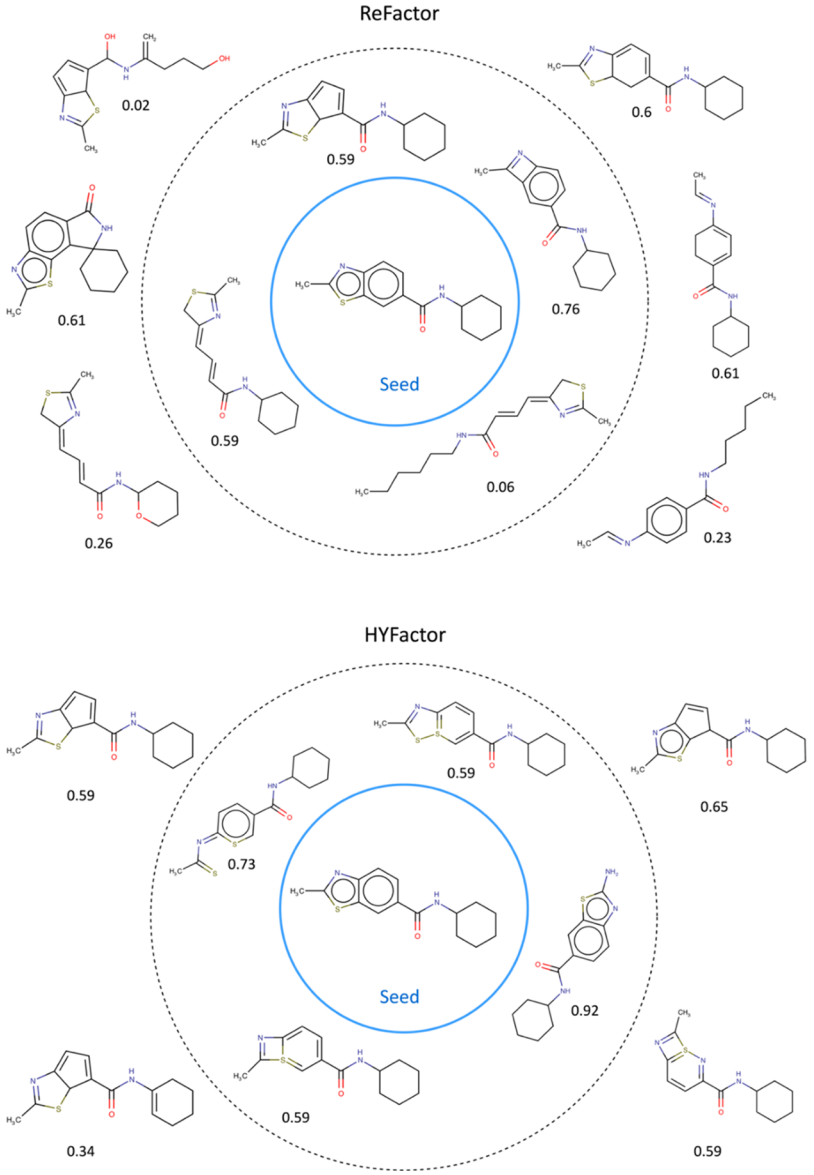


Figure 5. Example of structures generated with ReFactor and HyFactor trained on the MOSES set. Molecules generated with a standard deviation of <0.6 (>0.6) lie within (outside) the dashed circle. All molecules were aromatized with the ChemAxon toolbox. Each number corresponds to a pairwise Tanimoto similarity of a given generated structure with respect to the seed assessed with the atom-centered ISIDA fragments[35] involving sequences of atoms and bonds of sizes from 2 to 4 atoms with different labeling of cyclic and acyclic bonds.

https://doi.org/10.1021/acs.jcim.2c00744
J. Chem. Inf. Model. 2022, 62, 3524−3534

Table 4. Training Results on the ChEMBL Data Set

			number of training parameters (M)				reconstruction rate (%)	
architecture	batch	vector length	encoder	decoder	time per epoch[a] (min)	GPU memory[a] (MB)	training set	test set
ReFactor	1024	1024	35.7	14.8	~24.3	~ 22,845	99.8	95.2
HyFactor	1024	1024	25.3	15.1	~16.5	~ 16,755	99.7	95.0

[a]Measured in a "mixed precision" mode, which is available in the TensorFlow package. In this mode, a 16-bit floating-point type is used where it is possible; otherwise, a 32-bit floating-point type is applied.

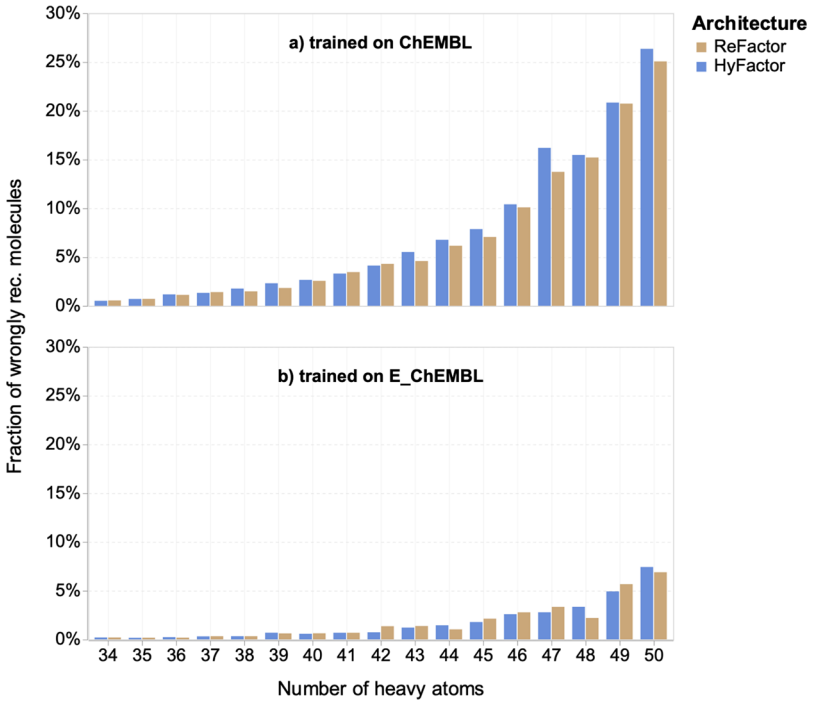


Figure 6. Distributions of errors in the ChEMBL validation set as a function of molecular size for ReFactor (gold) and HyFactor (blue) architectures trained on the (a) ChEMBL data set and (b) E_ChEMBL data set.

nonetheless represent correct structures. Valid structures generated with a high standard deviation parameter (STD ≥ 0.6) are characterized by high novelty and uniqueness. The validity of structures generated with HyFactor sharply drops for high STD values, which is not the case of ReFactor. Thus, one can suggest that the HyFactor latent space is more discontinuous than that of ReFactor. We notice that the latent space discontinuity is a quite common phenomenon for regular autoencoders without special regularization on latent space, like in variational autoencoders[33] or application of latent vectors for additional task solving.[34]

However, both architectures achieved high reconstruction rates and reasonable values of the validity, uniqueness, and novelty parameters at STD = 0.4 (Table 3). To compare with other autoencoders, we used this standard deviation. The results of MOSES benchmarking are given in Table 3, which includes the most important metrics assessing the generation ability of the proposed architectures. The results for all other metrics available in the MOSES package are given in the Supporting Information (see Table S3).

Since graphs are discrete objects, they occupy particular positions in the continuous autoencoder latent space. The latent vectors corresponding to invalid molecular graphs (e.g., disconnected structures or structures with valence mistakes) in the latent space are located between positions of valid molecules. While the proposed sampling method allows systematic exploration of the chemical space, the validity of the generated structures can hardly be controlled. Uniqueness and novelty are lower than for other architectures as we generate molecules in the vicinity of seed molecules.

Table 3 shows that the scaffold similarity (Scaf) of structures generated with HyFactor and ReFactor is lower than that for earlier reported autoencoders. Therefore, one can conclude that new architectures are more potent for scaffold hopping, which is crucial in de novo design. Moreover, HyFactor and ReFactor generate molecules with rather high internal diversity (IntDiv), characterizing a broader variety of generated structures. Compared to all benchmarked autoencoders, generated samples from ReFactor have the biggest Fréchet Chemnet distance (FCD) and the smallest similarity to a nearest neighbor (SNN). This demonstrates that the ReFactor's structures are more diverse with respect to the training set than those generated with any other architecture.

Another important issue concerns the neighborhood behavior analysis in the latent space. Thus, it is expected that for small STDs, the distances between generated latent vectors and the seed are rather small, which means that the generated chemical structures are similar to the seed structure. In order to check this hypothesis, for each of the 10K selected seed structures, several molecules were generated with HyFactor and ReFactor for standard deviations varying from 0.3 to 1.0 with a step of 0.02. At each step, 10 molecules were

https://doi.org/10.1021/acs.jcim.2c00744
J. Chem. Inf. Model. 2022, 62, 3524−3534

generated followed by the validity and uniqueness check. These simulations resulted, on average, in 10 and 40 generated structures per seed for HyFactor and ReFactor, respectively.

Generally, the neighborhood behavior is respected, that is, most of the structures generated with a small standard deviation (STD < 0.6) are more similar to the seed than those generated with a large STD (Figure 5). However, even with small STDs, ReFactor may occasionally generate very dissimilar structures corresponding to open-chain analogues of the cyclic seed structure. Such dissimilarity explains high FCD and low SNN scores observed for ReFactor compared to HyFactor and other considered architectures.

Data Augmentation: Case Study of the ChEMBL Database. The Achilles' heel of graph-based autoencoders is the reconstruction of molecules with a large number of atoms. Indeed, the probability of error in predicting the atom or bond type increases with molecular size. In this section, we demonstrate how data augmentation may help solve this problem. The experiments with ReFactor and HyFactor were performed on the ChEMBL database containing molecules bearing up to 50 heavy atoms. The results of training are given in Table 4 and specifications of training parameters are reported in the Supporting Information (Table S2).

According to Table 4, HyFactor uses 20% fewer training parameters than ReFactor in order to achieve a similar reconstruction rate, and thus, its training is 33% faster than ReFactor. Although the overall reconstruction rate of both networks is high enough, the reconstruction error sharply increases for molecules containing more than 35 atoms and it reaches almost 30% for ReFactor and HyFactor for molecules containing 50 atoms (Figure 6a). The latter can be explained by the small number of heavy molecules present in the training set (see Figure 1).

In order to confirm these suggestions, 460K virtual structures containing >35 atoms were generated using the Synt-On tool (former SynthI)[36] and then added to the ChEMBL set. These structures were generated using a special protocol insisting their similarity to related heavy ChEMBL molecules and synthetic feasibility; see details in the Supporting Information. The enriched ChEMBL set (E_ChEMBL) was then divided into training and test sets in the ratio 4:1 containing 1.6M and 420K structures, respectively. The distribution of molecular size in the obtained data set is given in Figure 1. Both architectures trained on the E_ChEMBL training set achieved reconstruction rates of >95% measured on the E_ChEMBL test set. The enrichment of the initial data set significantly reduced the reconstruction error of heavy molecules: for the molecules containing 50 atoms from ChEMBL, this value drops from some 25% for the models trained on ChEMBL (Figure 6a) to around 7% for the models trained on E_ChEMBL.

CONCLUSIONS

Neural network architecture HyFactor, which uses a hydrogen-count labeled graph as a chemical structure representation, has been developed. In this graph, implicit hydrogen atom counts are used instead of bond types, like in the InChI linear notation. Such a representation allows avoiding most of the problems of molecule standardization (aromatization, functional group standardization, and representation of Kekule structures) that make HyFactor insensible to molecule representation specifics.

For the sake of comparison, we have implemented the ReFactor architecture based on a classical molecular graph. It represents an updated and optimized version of the previously published DEFactor architecture, which uses defactorization of the adjacency matrix for graph generation. Since the latter proceeds in a single-shot manner without autoregressive bond and atom addition, the architecture is time and resource-economic.

Both ReFactor and HyFactor networks demonstrated high (>90%) reconstruction rates both in ZINC250K and ChEMBL datasets, which is similar to (or even better than) earlier reported graph-based or SMILES-based approaches. While the HyFactor architecture achieved the same reconstruction rate as ReFactor, it was also more effective in terms of the network parameters and training time.

Analysis of the dependency of reconstruction rates of HyFactor and ReFactor on molecular size revealed that, for molecules containing more than 35 atoms, the fraction of errors starts to grow, achieving almost 25% for molecules with 50 atoms. We hypothesized that this was related to the underrepresentation of such large molecules in the training dataset. Significant loss on the error rate on large molecules to 7% for the models trained on the dataset enriched by rationally generated virtual molecules supported this hypothesis. We believe that such a problem can be common for other graph- or SMILES-based autoencoders and encourage adding corresponding tests in generative chemistry benchmarking tools.

Both HyFactor and ReFactor can be used for efficient generation of new chemical structures. Since no special regularization of the latent space was used, vectors corresponding to new structures have been sampled around selected training set molecules. Novelty and uniqueness of generated structures increase as a function of the noise level, but the structures' validity drops in the same direction. In the MOSES benchmark, the structures generated by proposed architectures are characterized by greater diversity and scaffold novelty compared to those generated with the help of some other state-of-the-art approaches. This makes the proposed approaches especially promising for constrained molecule generation.

ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jcim.2c00744.

Detailed information about data sets used and some complementary modeling results (PDF)

AUTHOR INFORMATION

Corresponding Authors

Timur Madzhidov − *Laboratory of Chemoinformatics and Molecular Modeling, Butlerov Institute of Chemistry, Kazan Federal University, 420008 Kazan, Russia;* orcid.org/0000-0002-3834-6985; Email: Timur.Madzhidov@kpfu.ru

Alexandre Varnek − *Laboratory of Chemoinformatics, UMR 7140 CNRS, University of Strasbourg, 67081 Strasbourg, France;* orcid.org/0000-0003-1886-925X; Email: varnek@unistra.fr

https://doi.org/10.1021/acs.jcim.2c00744
J. Chem. Inf. Model. 2022, 62, 3524−3534

Authors

Tagir Akhmetshin − *Laboratory of Chemoinformatics, UMR 7140 CNRS, University of Strasbourg, 67081 Strasbourg, France;* orcid.org/0000-0002-2549-6431

Arkadii Lin − *Laboratory of Chemoinformatics, UMR 7140 CNRS, University of Strasbourg, 67081 Strasbourg, France*

Daniyar Mazitov − *Laboratory of Chemoinformatics and Molecular Modeling, Butlerov Institute of Chemistry, Kazan Federal University, 420008 Kazan, Russia*

Yuliana Zabolotna − *Laboratory of Chemoinformatics, UMR 7140 CNRS, University of Strasbourg, 67081 Strasbourg, France*

Evgenii Ziaikin − *Laboratory of Chemoinformatics and Molecular Modeling, Butlerov Institute of Chemistry, Kazan Federal University, 420008 Kazan, Russia;* orcid.org/0000-0001-6316-1301

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jcim.2c00744

Notes

The authors declare no competing financial interest.

The source code of HyFactor and all models obtained in this study are publicly available from our GitHub repository: https://github.com/Laboratoire-de-Chemoinformatique/HyFactor.

■ ACKNOWLEDGMENTS

T.A. thanks the Region Grand Est. for the Ph.D. fellowship.

■ REFERENCES

(1) Varnek, A.; Baskin, I. Machine Learning Methods for Property Prediction in Chemoinformatics: Quo Vadis? *J. Chem. Inf. Model.* **2012**, *52*, 1413−1437.

(2) Button, A.; Merk, D.; Hiss, J. A.; Schneider, G. Automated de Novo Molecular Design by Hybrid Machine Intelligence and Rule-Driven Chemical Synthesis. *Nat. Mach. Intell.* **2019**, *1*, 307−315.

(3) Segler, M. H. S.; Preuss, M.; Waller, M. P. Planning Chemical Syntheses with Deep Neural Networks and Symbolic AI. *Nature* **2018**, *555*, 604−610.

(4) Gómez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernández-Lobato, J. M.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; Aspuru-Guzik, A. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Cent. Sci.* **2018**, *4*, 268−276.

(5) Baskin, I. I. The Power of Deep Learning to Ligand-Based Novel Drug Discovery. *Expert Opin. Drug Discovery* **2020**, *15*, 755−764.

(6) Simonovsky, M.; Komodakis, N. GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; 2018; Vol. *11139* LNCS, pp. 412−422. DOI: 10.1007/978-3-030-01418-6_41.

(7) Kipf, T. N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *5th Int. Conf. Learn. Represent., ICLR 2017 - Conf. Track Proc.* 2017, 1−14. DOI: 10.48550/arXiv.1609.02907.

(8) Samanta, B.; De, A.; Jana, G.; Gómez, V.; Chattaraj, P. K.; Ganguly, N.; Gomez-Rodriguez, M. NeVAE: A Deep Generative Model for Molecular Graphs. *Proc. AAAI Conf. Artif. Intell.* **2019**, *33*, 1110−1117.

(9) Bresson, X.; Laurent, T. A Two-Step Graph Convolutional Decoder for Molecule Generation. *arXiv* **2019**, *1906*, No. 03412.

(10) Zhou, Z.; Kearnes, S.; Li, L.; Zare, R. N.; Riley, P. Optimization of Molecules via Deep Reinforcement Learning. *Sci. Rep.* **2019**, *9*, 10752.

(11) Assouel, R.; Ahmed, M.; Segler, M. H.; Saffari, A.; Bengio, Y. DEFactor: Differentiable Edge Factorization-Based Probabilistic Graph Generation. *arXiv* **2018**, 1−14.

(12) Jin, W.; Barzilay, R.; Jaakkola, T. Chapter 11. Junction Tree Variational Autoencoder for Molecular Graph Generation. In *RSC Drug Discovery Series*; 2020; pp. 228−249. DOI: 10.1039/9781788016841-00228.

(13) Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735−1780.

(14) Heller, S. R.; McNaught, A.; Pletnev, I.; Stein, S.; Tchekhovskoi, D. InChI, the IUPAC International Chemical Identifier. *J. Cheminf.* **2015**, *7*, 1−34.

(15) Kusner, M. J.; Paige, B.; Hernández-Lobato, J. M. Grammar Variational Autoencoder. *34th Int. Conf. Mach. Learn., ICML 2017* 2017, *4*, 3072−3084.

(16) Polykovskiy, D.; Zhebrak, A.; Sanchez-Lengeling, B.; Golovanov, S.; Tatanov, O.; Belyaev, S.; Kurbanov, R.; Artamonov, A.; Aladinskiy, V.; Veselov, M.; Kadurin, A.; Johansson, S.; Chen, H.; Nikolenko, S.; Aspuru-Guzik, A.; Zhavoronkov, A. Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models. *Front. Pharmacol.* **2020**, *11*, 1−10.

(17) Gaulton, A.; Bellis, L. J.; Bento, A. P.; Chambers, J.; Davies, M.; Hersey, A.; Light, Y.; McGlinchey, S.; Michalovich, D.; Al-Lazikani, B.; Overington, J. P. ChEMBL: A Large-Scale Bioactivity Database for Drug Discovery. *Nucleic Acids Res.* **2012**, *40*, 1100−1107.

(18) ChemAxon Ltd: Budapest, Hungary. https://chemaxon.com/ (accessed 2022-04-19).

(19) Bourlard, N.; Morgan, H. Generalization and Parameter Estimation in Feedforward Nets: Some Experiments. In *Proceedings of the 2nd International Conference on Neural Information Processing Systems*; Touretzky, D., Ed.; Morgan-Kaufmann, 1989; pp. 630--637.

(20) Pocha, A.; Danel, T.; Podlewska, S.; Tabor, J.; Maziarka, L. Comparison of Atom Representations in Graph Neural Networks for Molecular Property Prediction. In *2021 International Joint Conference on Neural Networks (IJCNN)*; IEEE, 2021; pp. 1−8. DOI: 10.1109/IJCNN52387.2021.9533698.

(21) Nugmanov, R. I.; Mukhametgaleev, R. N.; Akhmetshin, T.; Gimadiev, T. R.; Afonina, V. A.; Madzhidov, T. I.; Varnek, A. CGRtools: Python Library for Molecule, Reaction, and Condensed Graph of Reaction Processing. *J. Chem. Inf. Model.* **2019**, *59*, 2516−2521.

(22) Simonovsky, M.; Komodakis, N. Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs. *Proceedings - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017* 2017, *2017-Janua*, 29−38. DOI: 10.1109/CVPR.2017.11.

(23) Zitnik, M.; Agrawal, M.; Leskovec, J. Modeling Polypharmacy Side Effects with Graph Convolutional Networks. *Bioinformatics* **2018**, *34*, i457−i466.

(24) GoogleResearch. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. DOI: 10.5281/zenodo.5949169.

(25) Ba, J. L.; Kiros, J. R.; Hinton, G. E. Layer Normalization. *arXiv* **2016**, DOI: 10.48550/arXiv.1607.06450.

(26) Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP 2014−2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2014; pp. 1724−1734. DOI: 10.3115/v1/d14-1179.

(27) He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on Imagenet Classification. In *Proceedings of the IEEE International Conference on Computer Vision*; IEEE, 2015; Vol. *2015* Inter, pp. 1026−1034. DOI: 10.1109/ICCV.2015.123.

(28) Zhuang, J.; Tang, T.; Ding, Y.; Tatikonda, S.; Dvornek, N.; Papademetris, X.; Duncan, J. S. AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients. *Adv. Neural Inf. Process. Syst.* **2020**, *2020*, 1−29.

(29) Sattarov, B.; Baskin, I. I.; Horvath, D.; Marcou, G.; Bjerrum, E. J.; Varnek, A. De Novo Molecular Design by Combining Deep Autoencoder Recurrent Neural Networks with Generative Topographic Mapping. *J. Chem. Inf. Model.* **2019**, *59*, 1182−1196.
(30) Yan, C.; Wang, S.; Yang, J.; Xu, T.; Huang, J. Re-Balancing Variational Autoencoder Loss for Molecule Sequence Generation. *Proc. 11th ACM Int. Conf. Bioinformatics, Comput. Biol. Heal. Informatics, BCB 2020* 2020. DOI: 10.1145/3388440.3412458.
(31) Alperstein, Z.; Cherkasov, A.; Rolfe, J. T. All SMILES Variational Autoencoder. *arXiv* **2019**. DOI: 10.48550/arXiv.1905.13343.
(32) Dai, H.; Tian, Y.; Dai, B.; Skiena, S.; Song, L. Syntax-Directed Variational Autoencoder for Structured Data. *6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc.* 2018, 1−17.
(33) Kingma, D. P.; Welling, M. Auto-Encoding Variational Bayes. *2nd Int. Conf. Learn. Represent. ICLR 2014 - Conf. Track Proc.* 2014, 1−14.
(34) Winter, R.; Montanari, F.; Noé, F.; Clevert, D. A. Learning Continuous and Data-Driven Molecular Descriptors by Translating Equivalent Chemical Representations. *Chem. Sci.* **2019**, *10*, 1692−1701.
(35) Varnek, A.; Fourches, D.; Horvath, D.; Klimchuk, O.; Gaudin, C.; Vayer, P.; Solov'ev, V.; Hoonakker, F.; Tetko, I.; Marcou, G. ISIDA - Platform for Virtual Screening Based on Fragment and Pharmacophoric Descriptors. *Curr. Comput.-Aided Drug Des.* **2008**, *4*, 191−198.
(36) Zabolotna, Y.; Volochnyuk, D. M.; Ryabukhin, S. V.; Gavrylenko, K.; Horvath, D.; Klimchuk, O.; Oksiuta, O.; Marcou, G.; Varnek, A. SynthI: A New Open-Source Tool for Synthon-Based Library Design. *J. Chem. Inf. Model.* **2022**, *62*, 2151−2163.