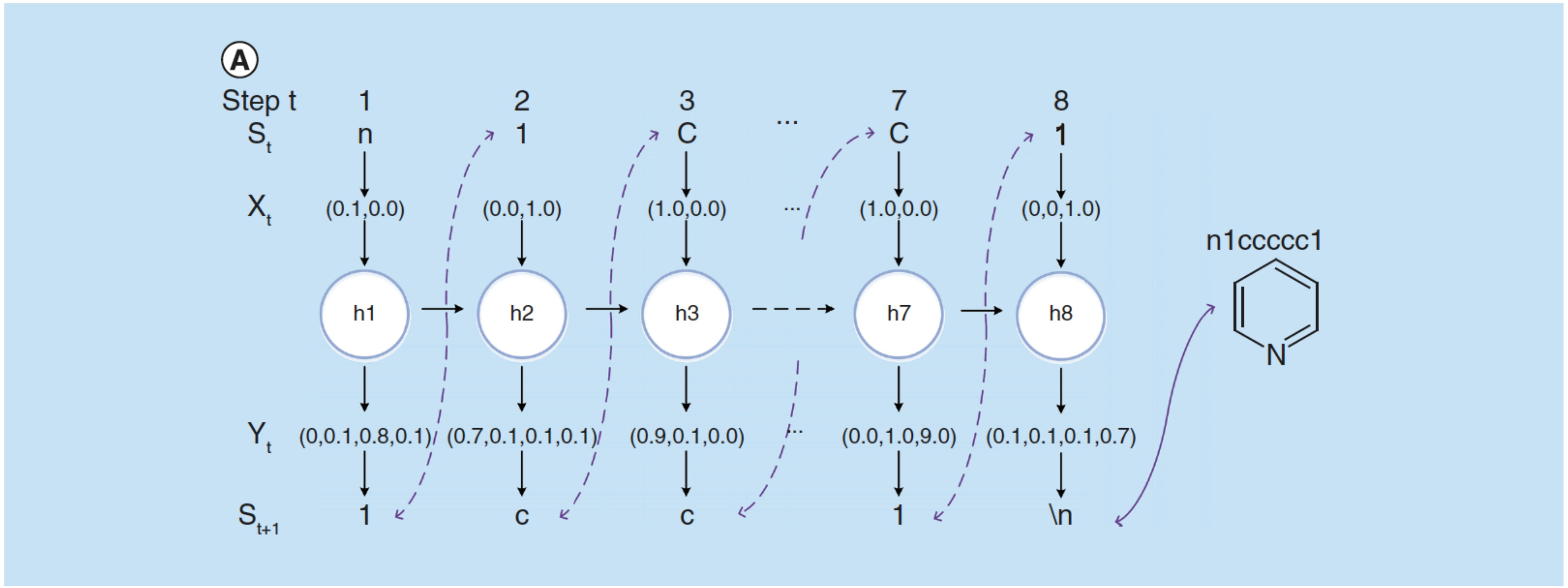
CS 6824: Deep Generative Learning for Molecular Synthesis

Acknowledgement:
Many of the images in the slides are derived from images.google.com or other publicly available sources.

Recurrent neural network



$$\boldsymbol{h}_i = f_r(\boldsymbol{h}_{i-1}, \boldsymbol{x}_i)$$

$$\boldsymbol{y}_i = f_o(\boldsymbol{h}_i)$$

- Standard RNN suffers from vanishing gradient problem
- RNN variants of long short-term memory (LSTM) and gated recurrent unit (GRU) aim to address that

https://doi.org/10.4155/fmc-2018-0358

RNN-based generative models with TL

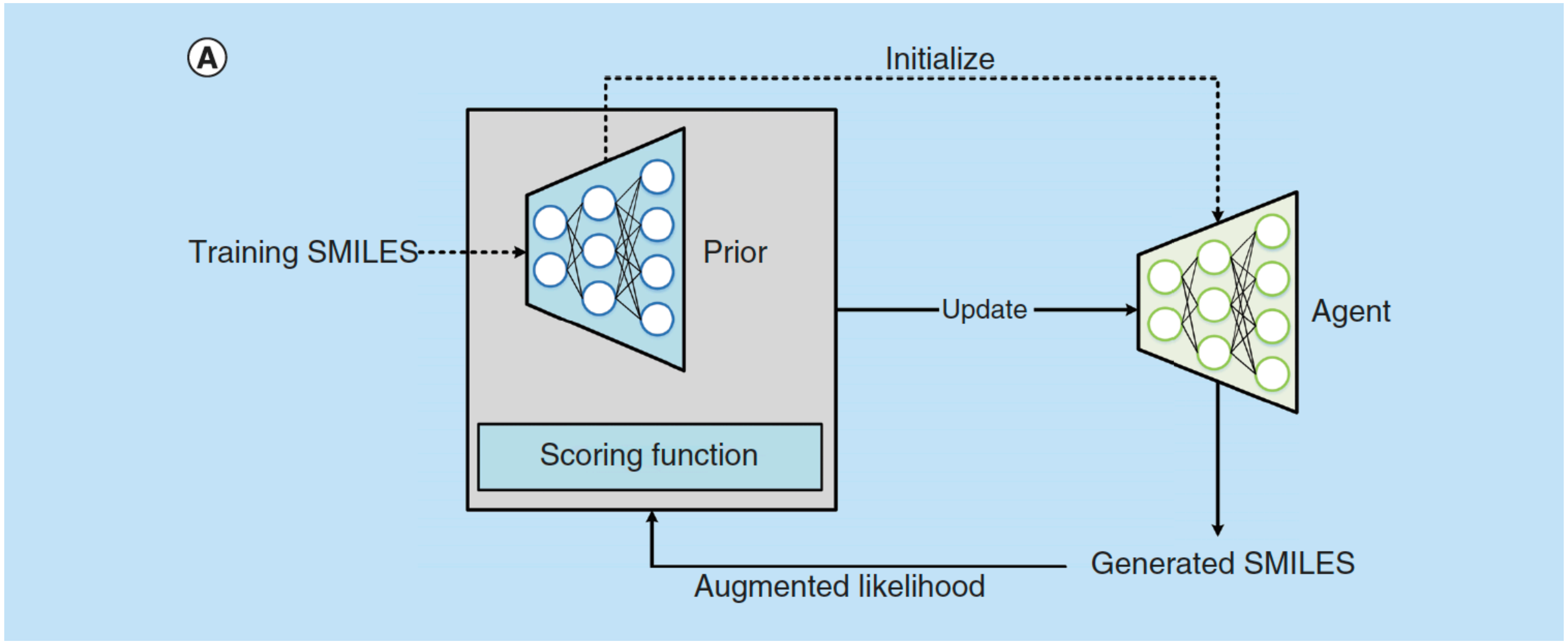
- RNN generative model for *de novo* molecular generation using stacked LSTM layer

- LSTM-based RNN model combined with a sampling temperature, which rescales the probability distribution of output sequences

https://doi.org/10.4155/fmc-2018-0358

RNN-based generative models with RL

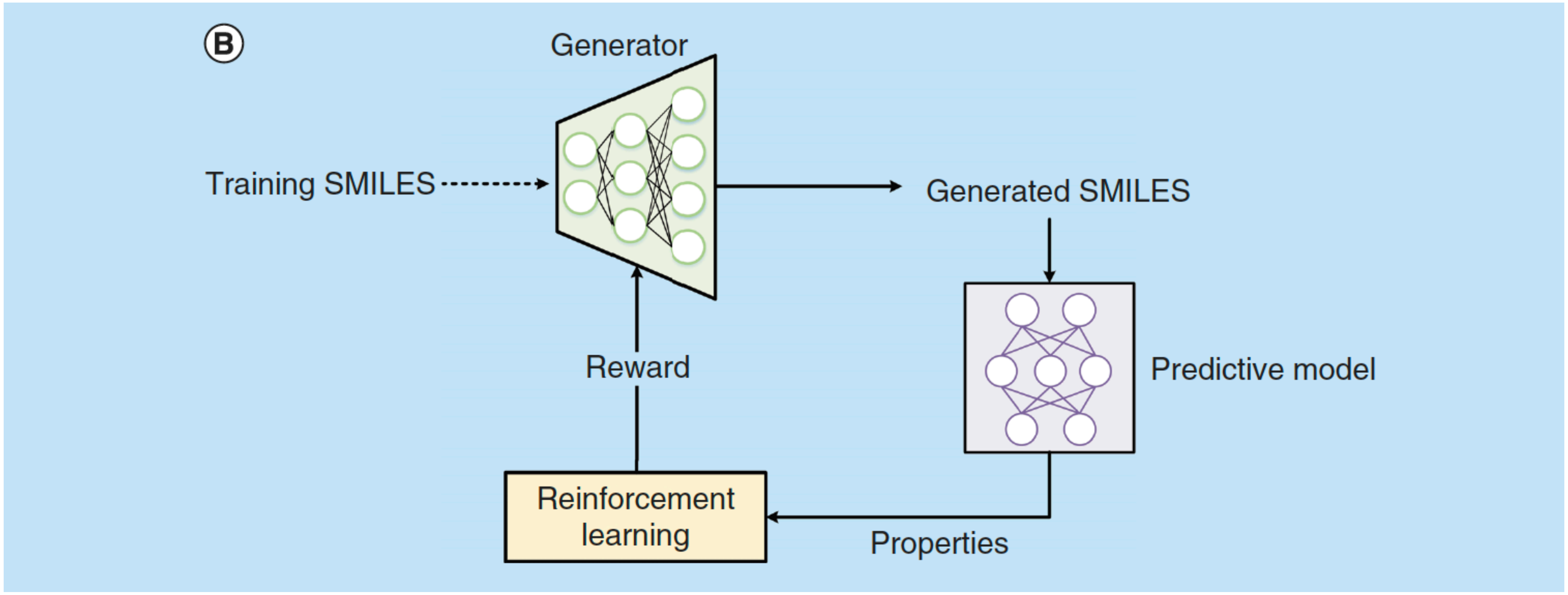
- **REINVENT**
  - RNN-based generative model for molecular *de novo* design through augmented episodic likelihood-based RL
  - Policy-based RL to fine-tune an RNN-based agent for generating molecules with given desirable properties



https://doi.org/10.4155/fmc-2018-0358

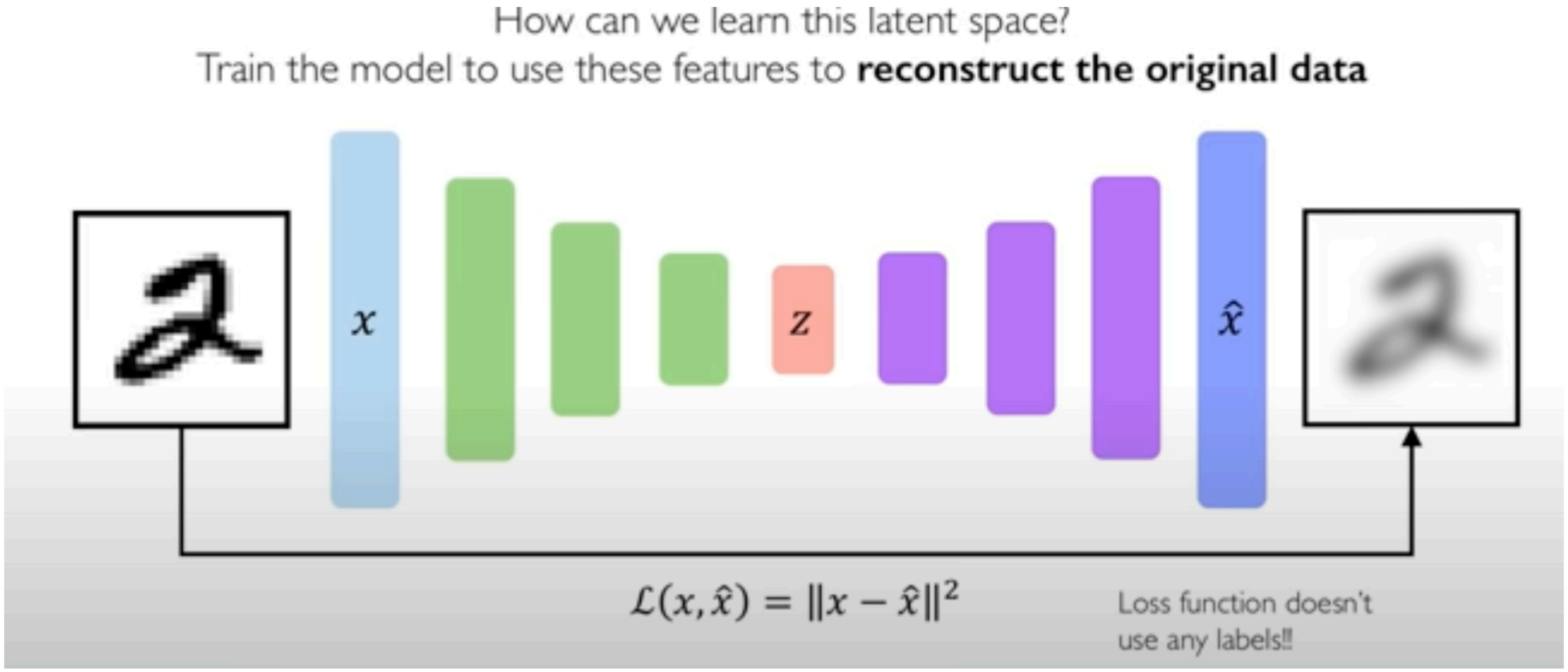
RNN-based generative models with RL

- **ReLeaSE**
  - Combines the two deep neural networks (generative model G and predictive model P trained separately
  - G is a stack-augmented RNN (Stack-RNN) architecture to learn hidden rules of forming sequences of letters for generating valid SMILES molecules
  - P is analogous to a Quantitative Structure-Activity Relationship (QSAR) model for molecular properties prediction with only taking SMILES string as an input vector. It is based on a deep neural network consisting of embedding layer, LSTM layer and two dense layers



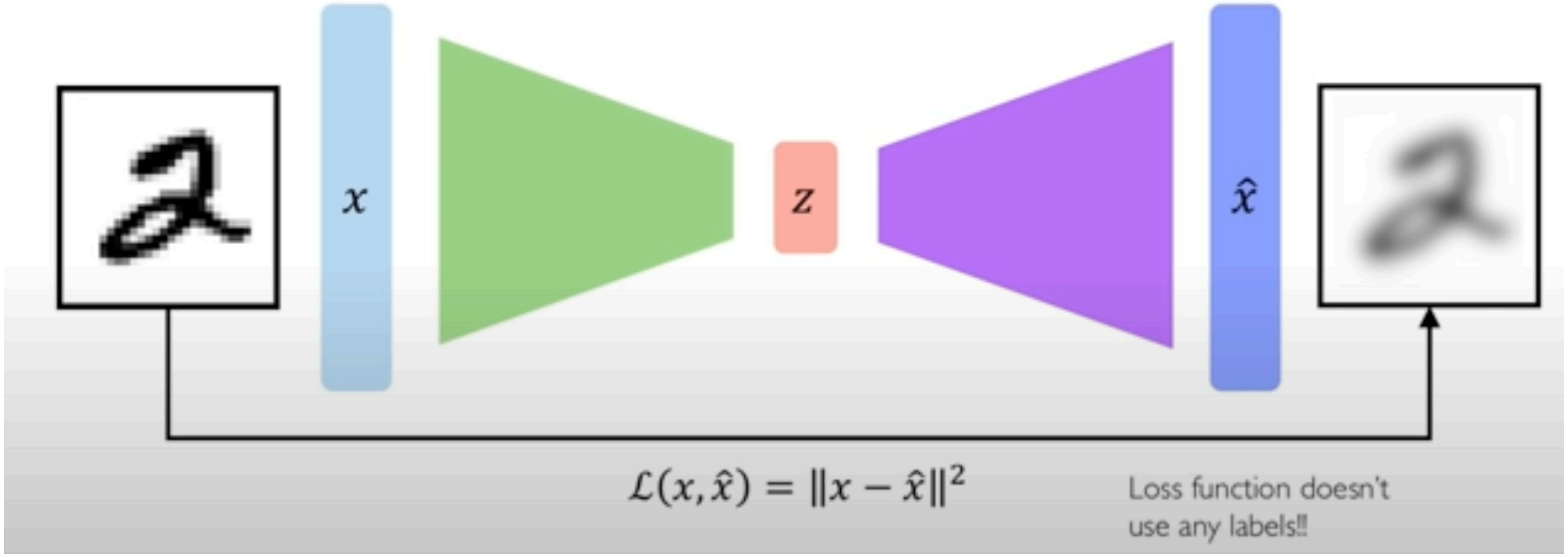
https://doi.org/10.4155/fmc-2018-0358

Autoencoders


How can we learn this latent space?
Train the model to use these features to **reconstruct the original data**
x
z
$\hat{x}$
$\mathcal{L}(x,\hat{x}) = \|x - \hat{x}\|^2$
Loss function doesn't use any labels!!

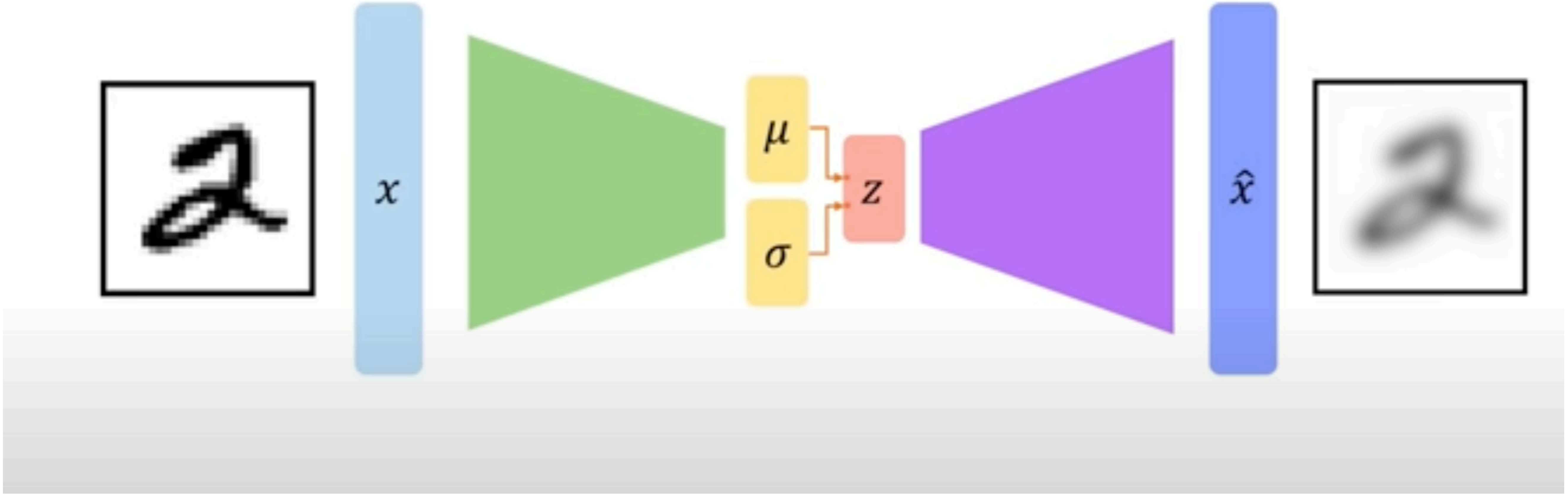
- **Autoencoding** = **Auto**matically **encoding** data

Traditional autoencoders


x
z
$\hat{x}$
$\mathcal{L}(x, \hat{x}) = \|x - \hat{x}\|^2$
Loss function doesn't use any labels!!

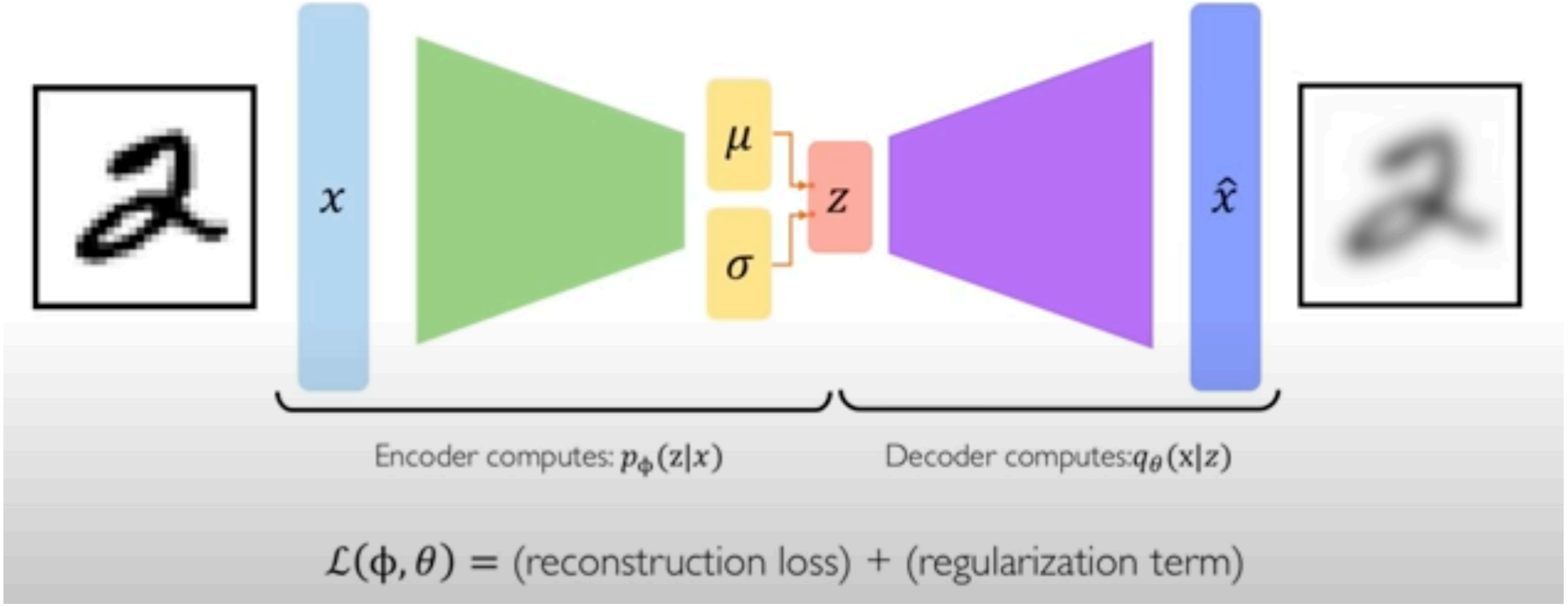
- Deterministic encoding

Variational autoencoders (VAEs)



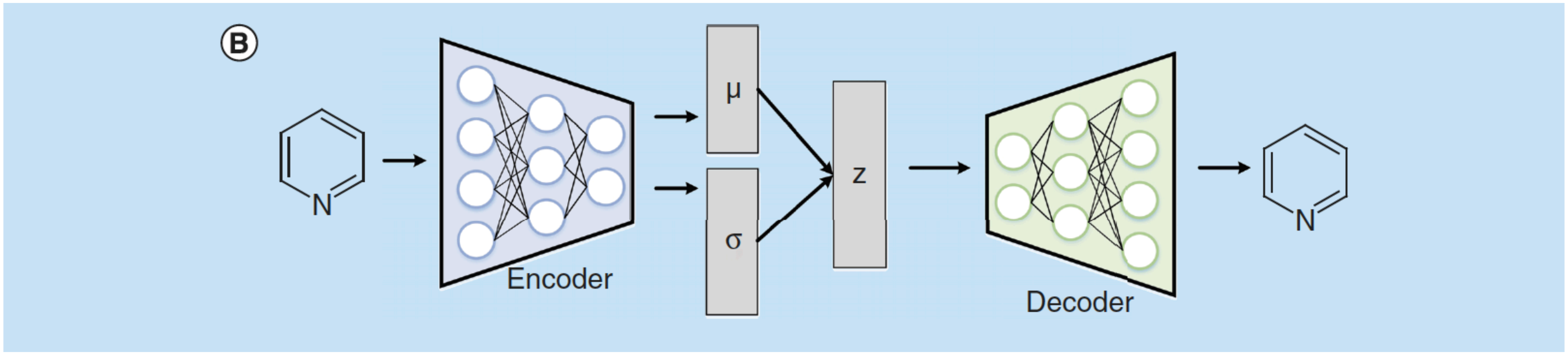
- Replace the deterministic bottleneck layer with a stochastic sampling operation

Variational autoencoders (VAEs)


x
μ
σ
z
$\hat{x}$
Encoder computes: $p_\phi(z|x)$
Decoder computes: $q_\theta(x|z)$
$\mathcal{L}(\phi, \theta)$ = (reconstruction loss) + (regularization term)

- Going from learning a vector of latent variables to a vector of means and variances which describe the prob. distribution associated with each of the latent variables
- Both encoder and decoder are probabilistic in nature

VAE for molecular data



$$\mathrm{ELBO}(\phi,\theta)=\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right]-\mathrm{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})\,||\,p(\boldsymbol{z})\right)$$

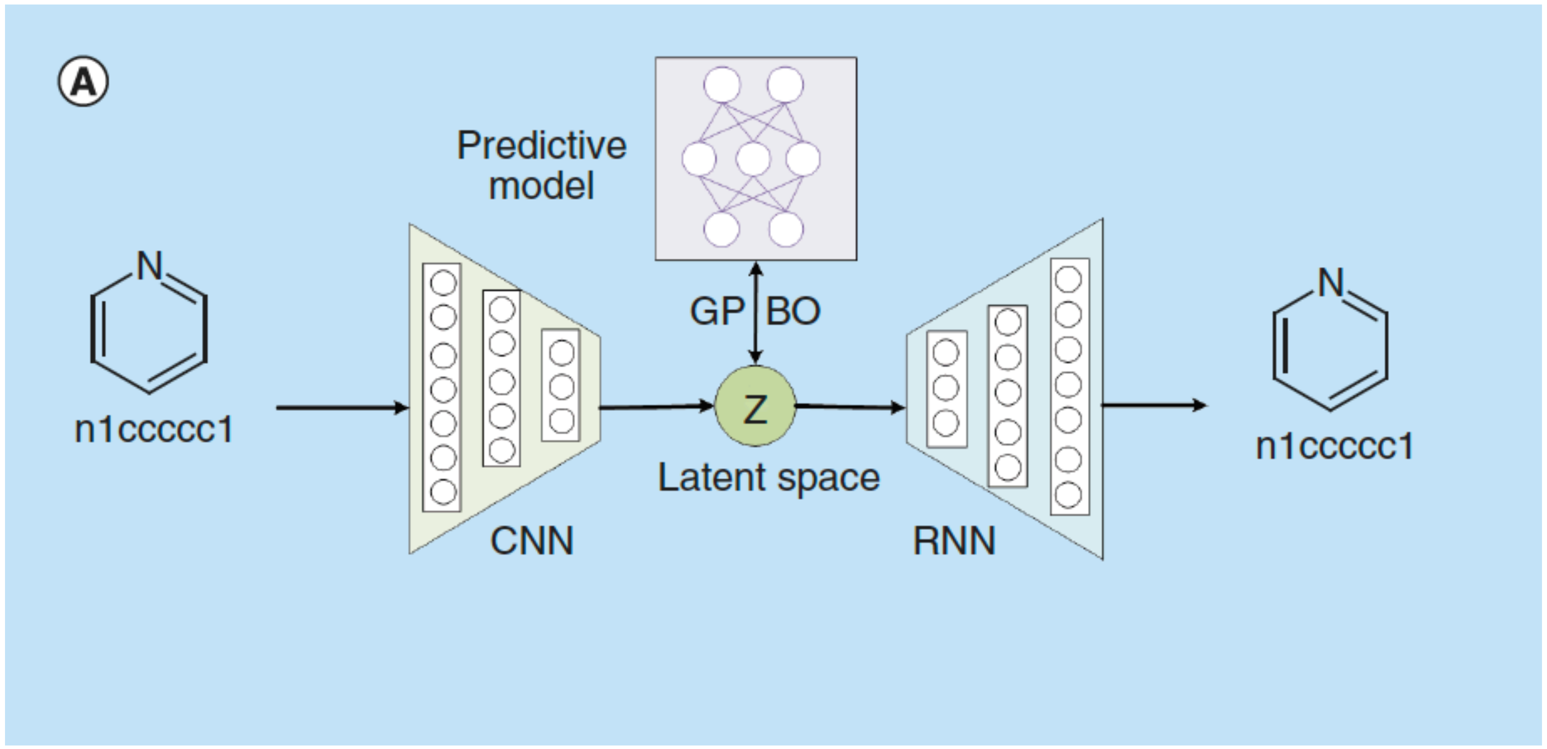
- **Encoder**: learning to represent molecules in a continuous manner that facilitates the prediction and optimization of their properties

- **Decoder**: learning to map an optimized continuous representation back into a molecular with improved properties

https://doi.org/10.4155/fmc-2018-0358

RNN- and AE-based generative models

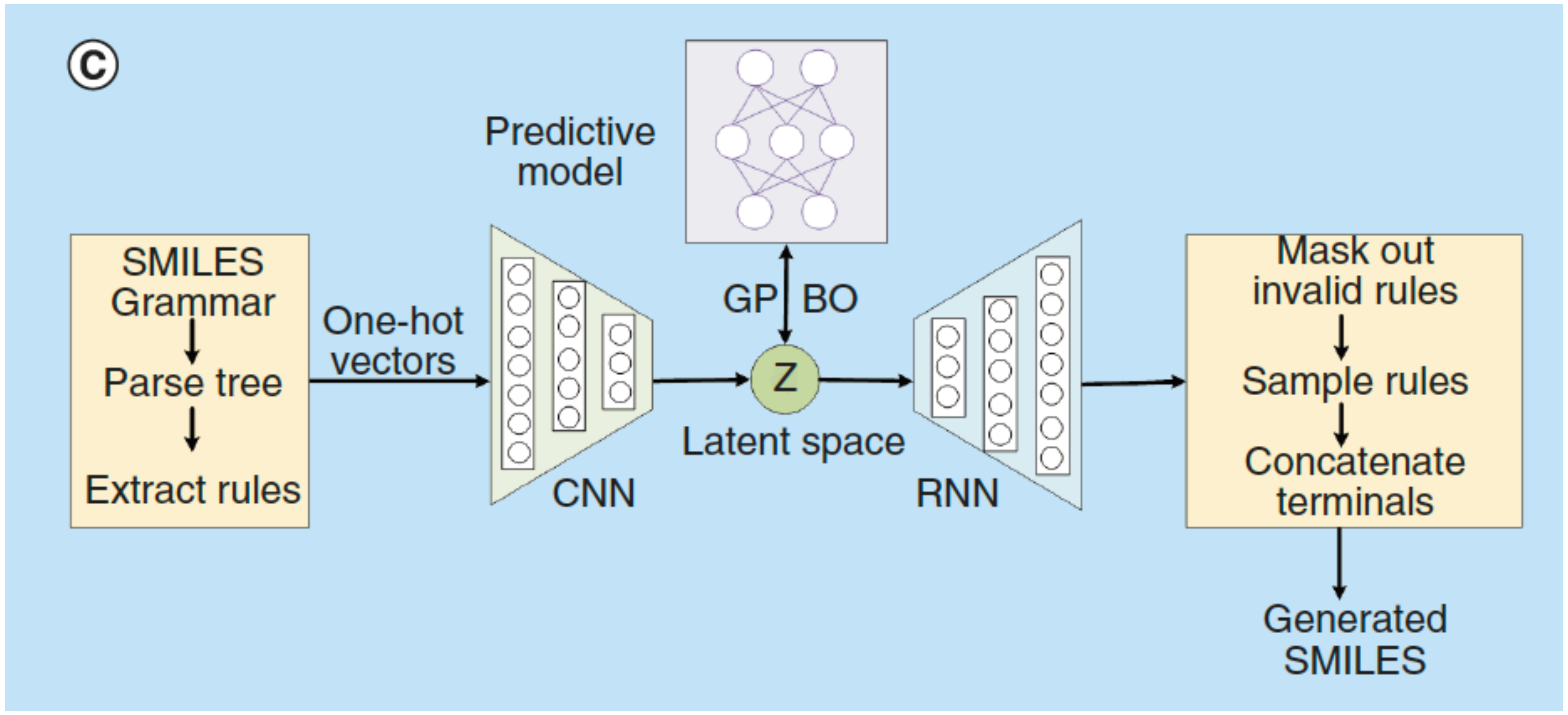
- **ChemVAE**
  - Encoder converts the discrete representations of molecules (SMILES strings in this case) into real-valued fix-dimensional continuous vectors
  - Decoder transforms the vectors to SMILES strings
  - Adds Gaussian noise to the encoder with penalty term guaranteeing the valid decoding
  - A predictive model based on multilayer perceptron, was joined into VAE to predict the molecular properties from latent space



https://doi.org/10.4155/fmc-2018-0358

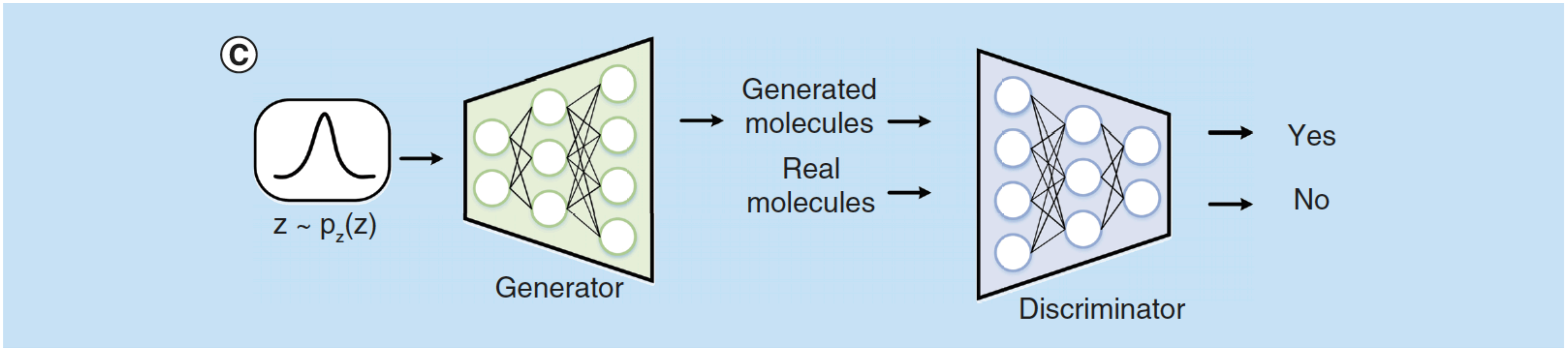
RNN- and AE-based generative models

- **Grammar VAE**
    - Utilizes a context-free grammar (CFG) to form a parse tree, which is decomposed into a sequence of production rules defined as 4-tuple G = (V, T, R, S), containing a finite set of nonterminal symbols V, a finite set of terminal symbols T, a finite set of production rules R, and a distinct start symbol S.
    - The SMILES strings can be generated via adopting production rules recursively (sampling from start symbol till no nonterminals left)
    - Rules are fed into an encoder with convolutional neural network architecture, then an RNN architecture as a decoder for generating syntactically valid SMILES
    - Decoder transforms the vectors into SMILES strings



https://doi.org/10.4155/fmc-2018-0358

Generative adversarial network



- A generative model G, which learns a map from a prior to the data distribution to sample new data points,
- discriminative model D, which learns to classify whether samples come from the real data distribution rather than from G
- Those two models are implemented as deep neural networks and trained alternatively with stochastic gradient descent. G and D have different objectives, and they can be seen as two players in a minmax game

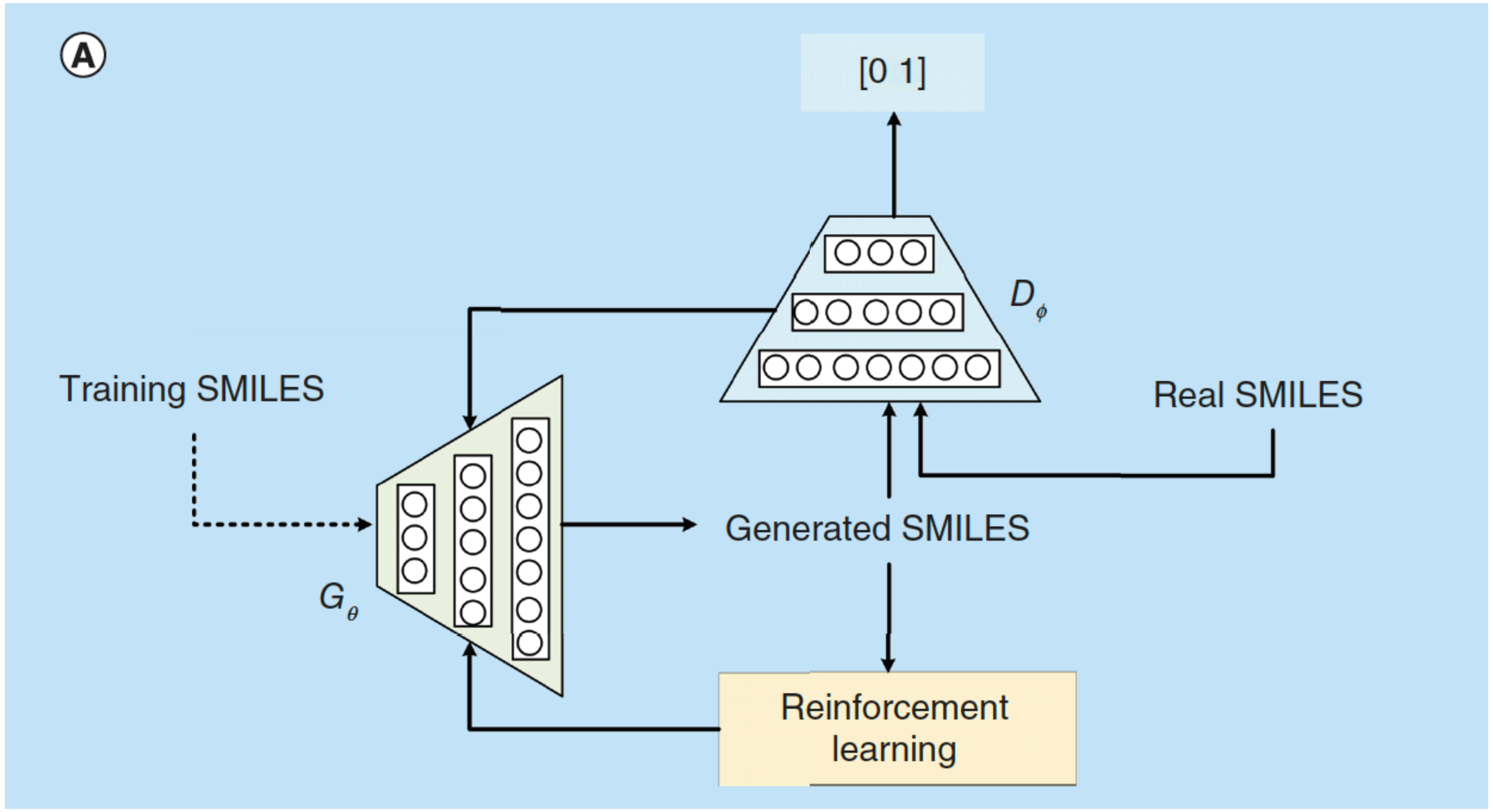
$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[\log D(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[\log(1 - D(G(\boldsymbol{z})))\right]$$

- G tries to generate samples to fool the discriminator and D tries to differentiate samples correctly

https://doi.org/10.4155/fmc-2018-0358

RNN & GAN-based generative models with RL

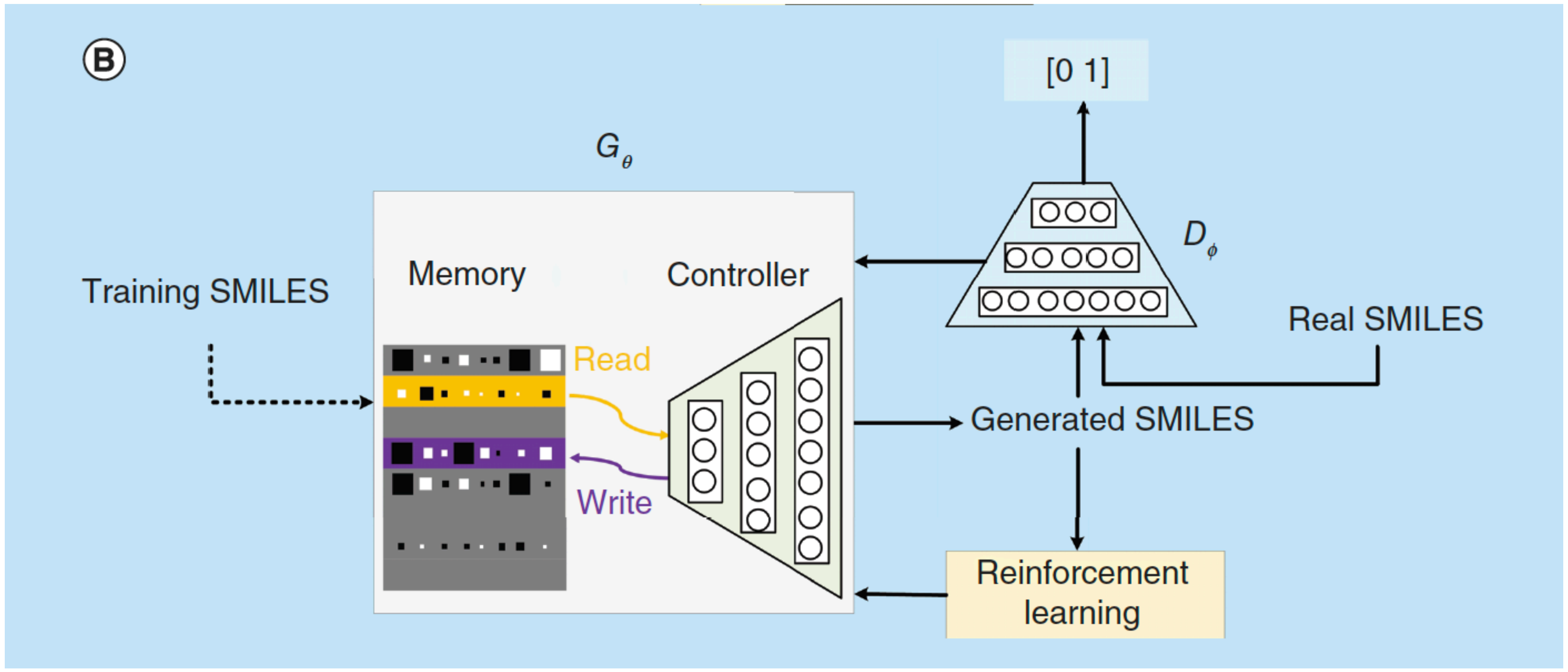
- **Objective reinforced GAN (ORGAN)**
  - GAN architecture is combined with reward functions with RL to generate SMILES strings
  - G_θ is a LSTM-based generator parameterized by θ, that produces high-quality sequences X1:T = (x1, ..., xT). And a discriminator D_ϕ parameterized by ϕ is a convolutional neural network specifically for sequence classification
  - G_θ is trained as an agent and the reward function R was supplied by Dϕ to fool Dϕ
  - D_ϕ is trained to classify real and generated SMILES sequences



https://doi.org/10.4155/fmc-2018-0358

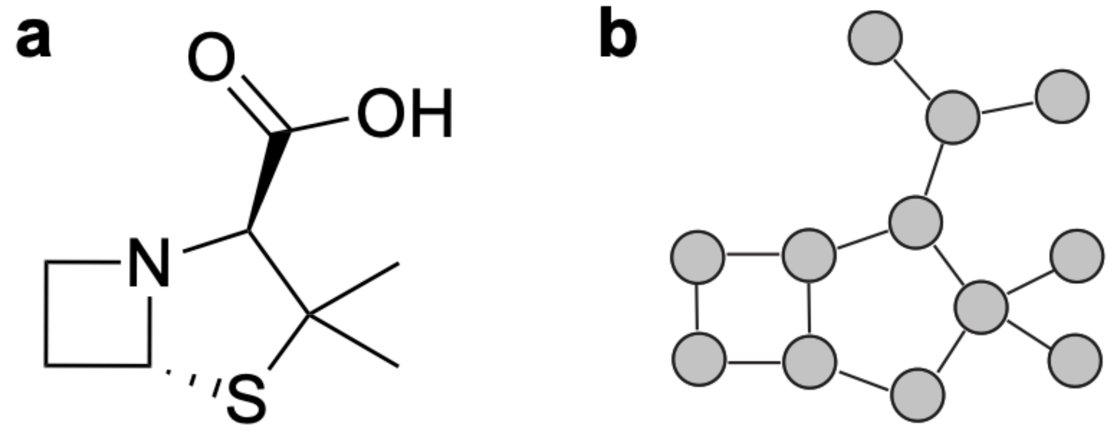
RNN & GAN-based generative models with RL

- **Reinforced adversarial neural computer**
  - generator G_θ is a differentiable neural computer (DNC), which is is an LSTM controller with external
  - memory (like Stack-RNN) and its advantages lay in its powerful memory to reconstruct and generate complex and much longer SMILES strings than LSTM
  - The action value function of candidate states (partial sequences) was calculated by Monte Carlo search.



https://doi.org/10.4155/fmc-2018-0358

Graph-based generative models

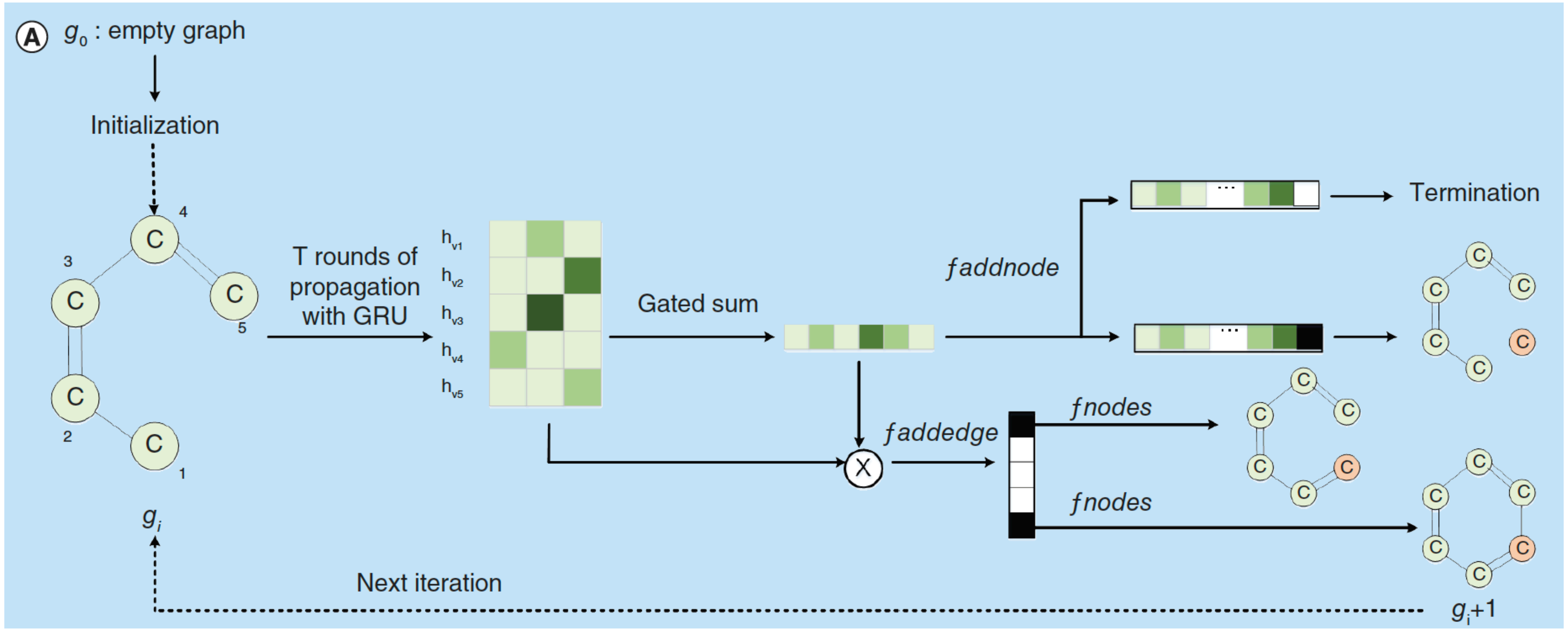


◦ Given an example of chemical molecule, a 2D structure is a graph with nodes as its atoms and edges between two nodes as its bonds

https://doi.org/10.4155/fmc-2018-0358

Recurrent graph-based generative models

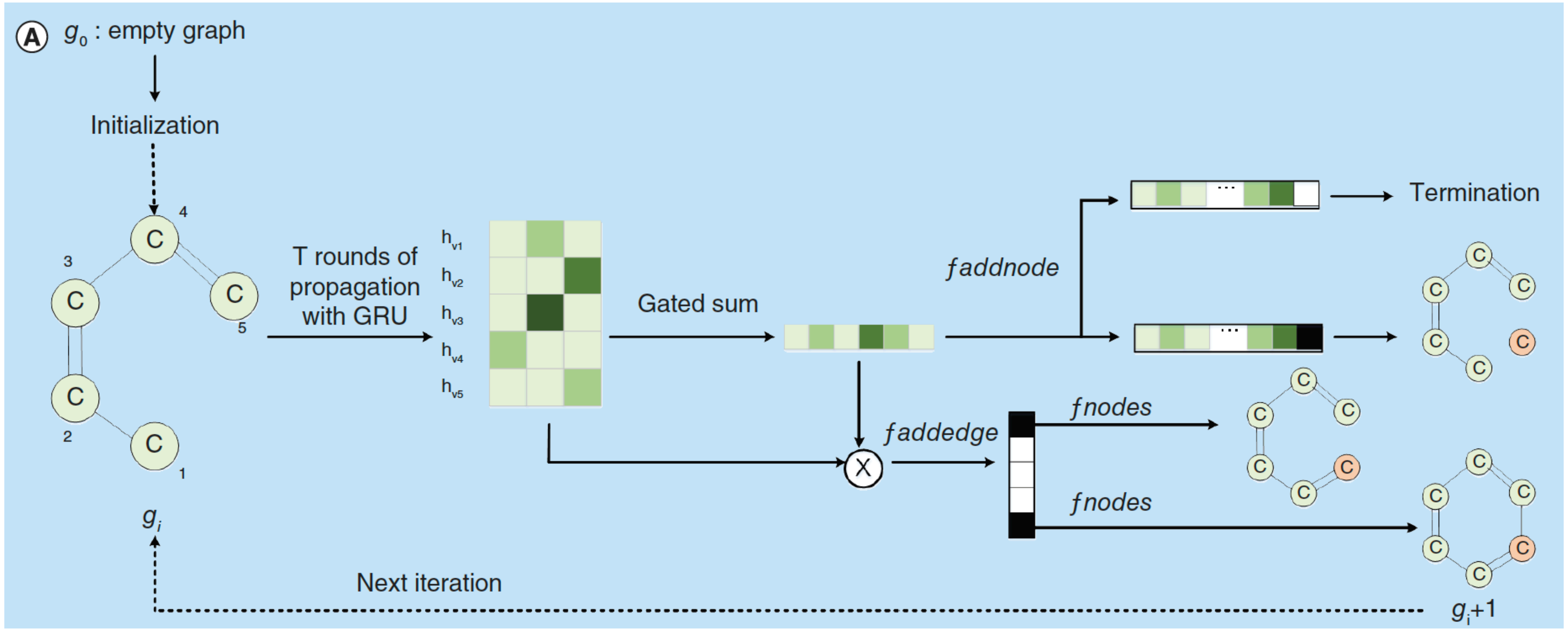
- **GraphNet**
  - Uses the structure of molecular graph to create representations of atoms and bonds via an information propagation process (GRU) ; these representations are used to make sequential graph building decisions
  - Probabilistic decision-making modules (parameterized by training with known molecular graphs)
    - adding a new node or not (with probabilities provided by a $f_{addnode}$ module)
    - adding a new edge or not (probabilities provided by $f_{addedge}$ module)
    - picking one node to connect to the new node with typed edges (probabilities provided by f_{nodes} module)



https://doi.org/10.4155/fmc-2018-0358

Recurrent graph-based generative models

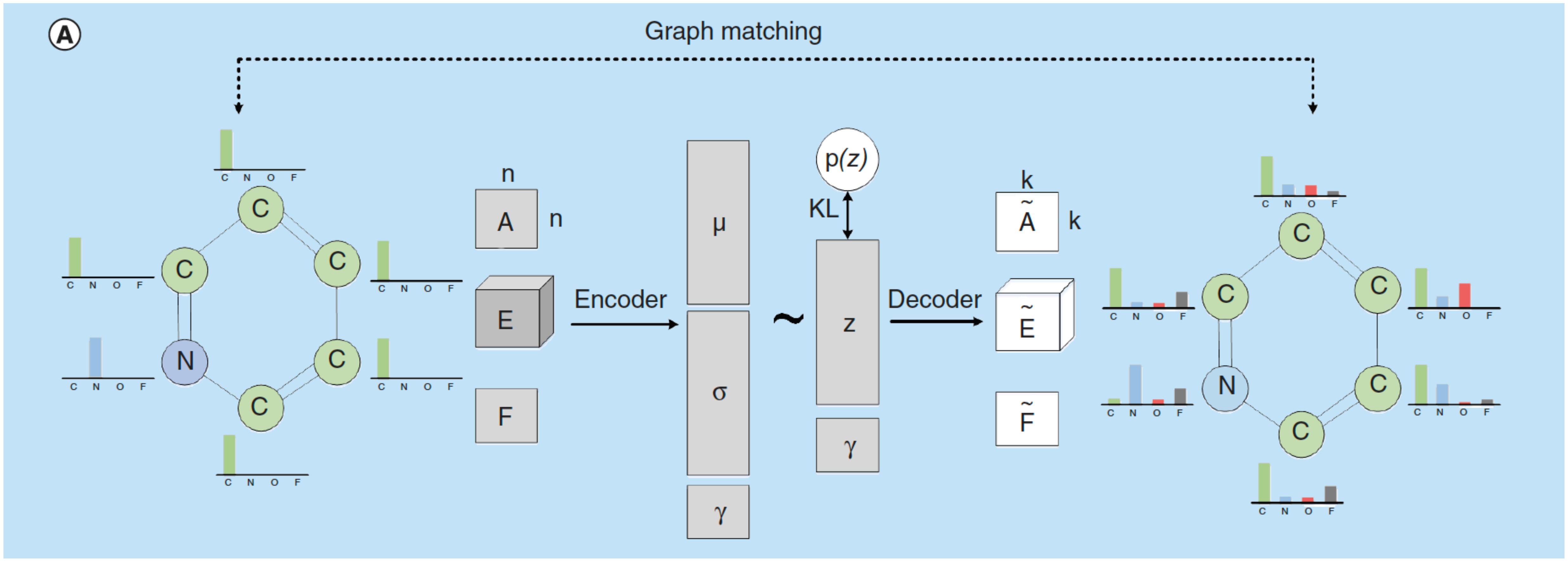
- **MolMP**
  - While GraphNet uses GRU to obtain atomic representations, which are integrated to molecular representation by Gated Sum,MolMP uses Graph Convolutional Network (GCN) and average pooling for atomic and molecular representations
  - The actions of $f_{addnode}$, $f_{addedge}$ and f_{nodes} are merged into a single f_{append} step, and $f_{connect}$ is used to avoid the repeated operation of adding edges. It helps to reduce the number of steps during generation.adding a new node or not



https://doi.org/10.4155/fmc-2018-0358

VAE-based generative models

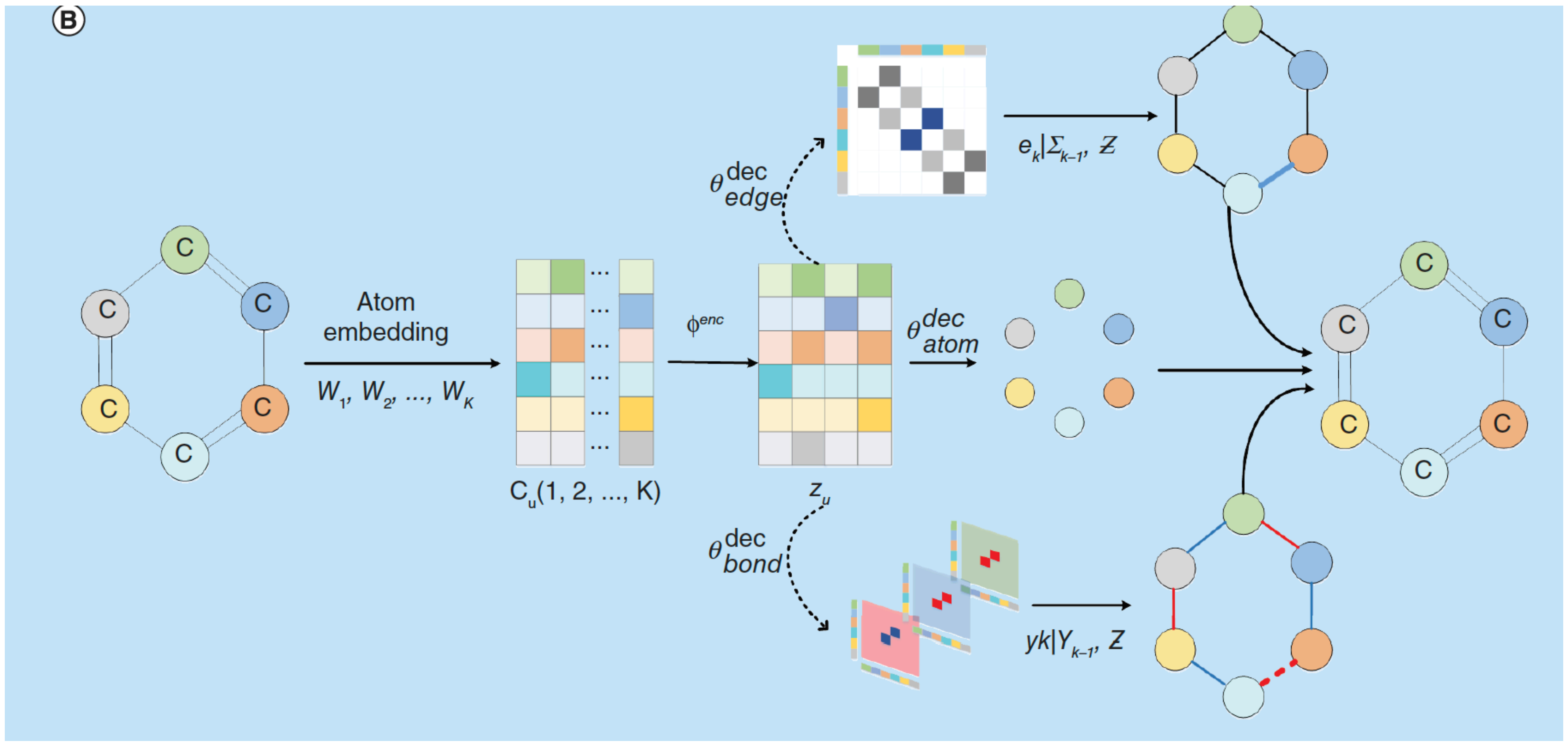
- **GraphVAE**
  - A molecular graph can be characterized by G = (A, E,F) with its adjacency matrix A, edge attribute tensor E, and node attribute tensor F
  - VAE was used to jointly train an encoder $q_\phi(z|G)$ and a decoder $p_\theta(G|z)$ to map between the space of graphs G and the continuous embedding $z \in R^D$, where ϕ and θ are learned parameters. A regularization term, KL-divergence, is added into the latent code space with a prior isotropic Gaussian distribution $p(z) = N(0, I)$, which is aimed to approximate the two distributions of $q\phi(z|G)$ and $p(z)$.



https://doi.org/10.4155/fmc-2018-0358

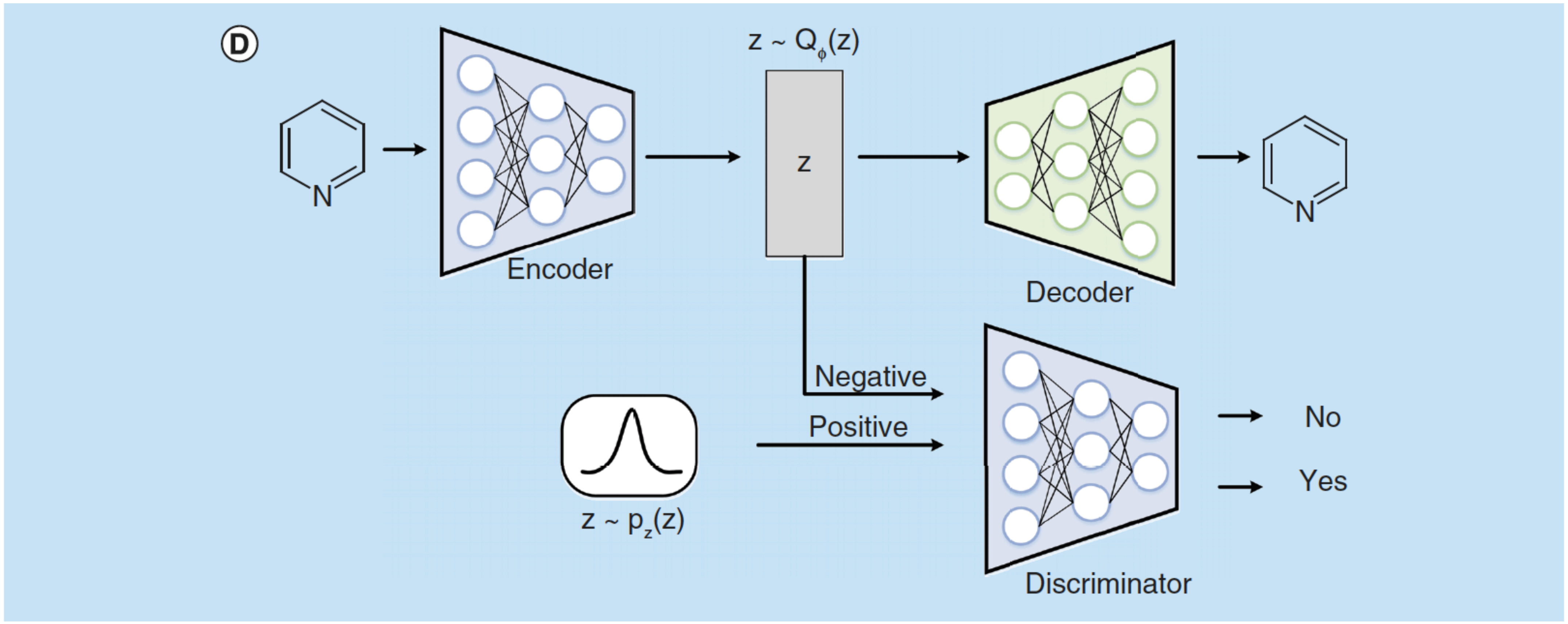
VAE & RNN-based generative models

- **NeVAE**
  - A step-wise generative model for undirected molecular graphs based on VAEs.
  - Defines a probabilistic encoding for each atom by extracting atomic information from K different layers
  - This information is fed into a neural network to make the product obey the stand normal distribution for each atom
  - This atom-based embedding strategy is invariant to permutations of the atoms and do not depend on the number of atoms and bonds, thus allowing for variable-sized molecular graphs
  - A whole molecular graph is decoded out with dynamic recurrent updating of the edges and edge weight



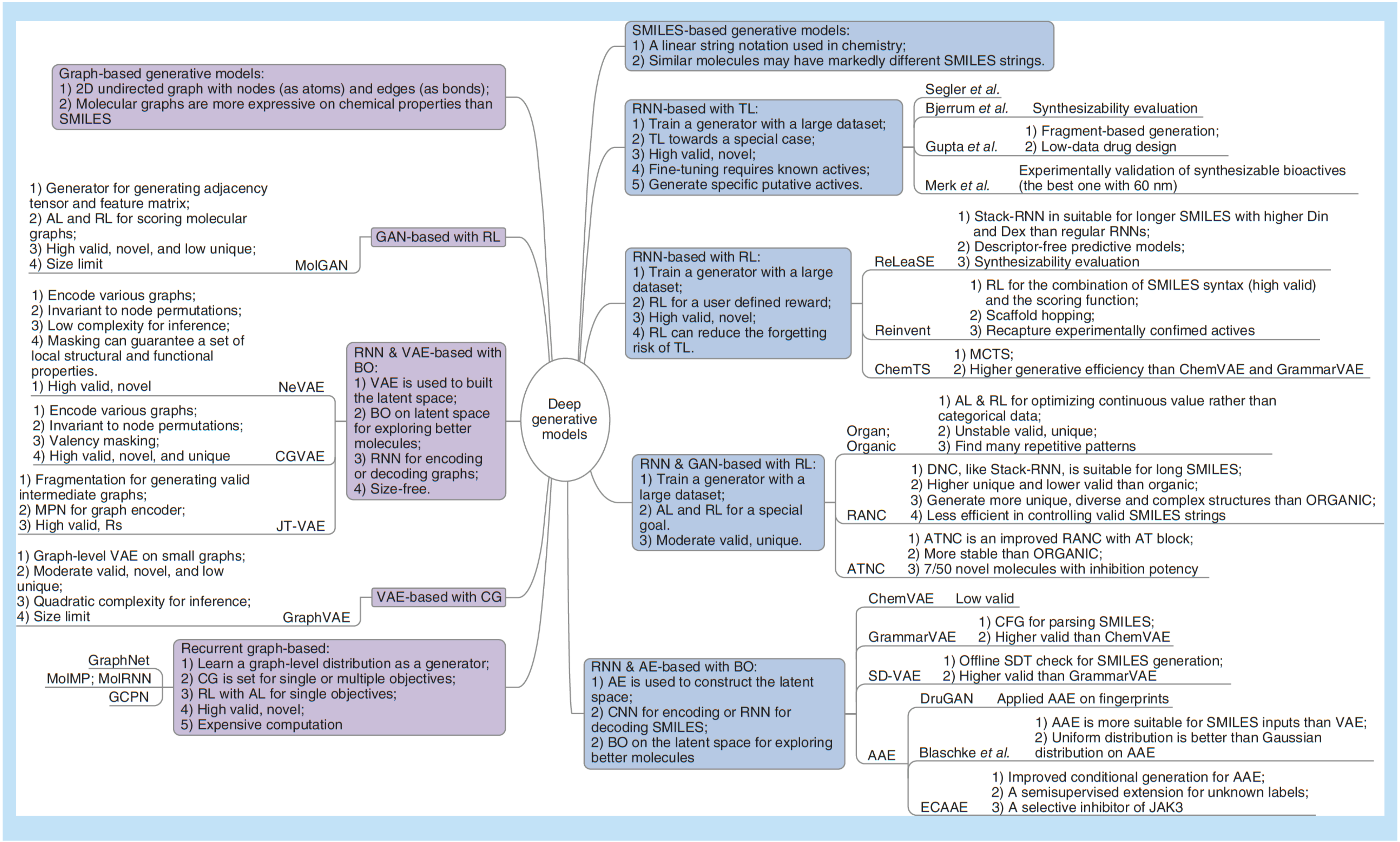
https://doi.org/10.4155/fmc-2018-0358

Adversarial autoencoder



- Inspired by VAE and GAN, AAE is proposed as a standard AE regularized by an adversarial learning (AL) procedure rather than a KL divergence penalty
- While the KL regularization in VAE is usually used to impose a prior distribution on the latent code z, the AL regularization in AAE is utilized to match the posterior distribution to a prior distribution
- While the posterior distribution in VAE is usually a Gaussian distribution with mean and variance predicted by the encoder, posterior distribution in AAE is encouraged to match a prior arbitrary distribution
- G tries to fool the discriminator D by mimicking the prior arbitrary distribution

https://doi.org/10.4155/fmc-2018-0358



Deep generative models

Graph-based generative models:
1) 2D undirected graph with nodes (as atoms) and edges (as bonds);
2) Molecular graphs are more expressive on chemical properties than SMILES

GAN-based with RL
- MolGAN: 1) Generator for generating adjacency tensor and feature matrix; 2) AL and RL for scoring molecular graphs; 3) High valid, novel, and low unique; 4) Size limit

RNN & VAE-based with BO:
1) VAE is used to built the latent space;
2) BO on latent space for exploring better molecules;
3) RNN for encoding or decoding graphs;
4) Size-free.
- NeVAE: 1) Encode various graphs; 2) Invariant to node permutations; 3) Low complexity for inference; 4) Masking can guarantee a set of local structural and functional properties. 1) High valid, novel
- CGVAE: 1) Encode various graphs; 2) Invariant to node permutations; 3) Valency masking; 4) High valid, novel, and unique
- JT-VAE: 1) Fragmentation for generating valid intermediate graphs; 2) MPN for graph encoder; 3) High valid, Rs

VAE-based with CG
- GraphVAE: 1) Graph-level VAE on small graphs; 2) Moderate valid, novel, and low unique; 3) Quadratic complexity for inference; 4) Size limit

Recurrent graph-based:
1) Learn a graph-level distribution as a generator;
2) CG is set for single or multiple objectives;
3) RL with AL for single objectives;
4) High valid, novel;
5) Expensive computation
- GraphNet
- MolMP; MolRNN
- GCPN

SMILES-based generative models:
1) A linear string notation used in chemistry;
2) Similar molecules may have markedly different SMILES strings.

RNN-based with TL:
1) Train a generator with a large dataset;
2) TL towards a special case;
3) High valid, novel;
4) Fine-tuning requires known actives;
5) Generate specific putative actives.
- Segler *et al.*
- Bjerrum *et al.*: Synthesizability evaluation
- Gupta *et al.*: 1) Fragment-based generation; 2) Low-data drug design
- Merk *et al.*: Experimentally validation of synthesizable bioactives (the best one with 60 nm)

RNN-based with RL:
1) Train a generator with a large dataset;
2) RL for a user defined reward;
3) High valid, novel;
4) RL can reduce the forgetting risk of TL.
- ReLeaSE: 1) Stack-RNN in suitable for longer SMILES with higher Din and Dex than regular RNNs; 2) Descriptor-free predictive models; 3) Synthesizability evaluation
- Reinvent: 1) RL for the combination of SMILES syntax (high valid) and the scoring function; 2) Scaffold hopping; 3) Recapture experimentally confimed actives
- ChemTS: 1) MCTS; 2) Higher generative efficiency than ChemVAE and GrammarVAE

RNN & GAN-based with RL:
1) Train a generator with a large dataset;
2) AL and RL for a special goal.
3) Moderate valid, unique.
- Organ; Organic: 1) AL & RL for optimizing continuous value rather than categorical data; 2) Unstable valid, unique; 3) Find many repetitive patterns
- RANC: 1) DNC, like Stack-RNN, is suitable for long SMILES; 2) Higher unique and lower valid than organic; 3) Generate more unique, diverse and complex structures than ORGANIC; 4) Less efficient in controlling valid SMILES strings
- ATNC: 1) ATNC is an improved RANC with AT block; 2) More stable than ORGANIC; 3) 7/50 novel molecules with inhibition potency

RNN & AE-based with BO:
1) AE is used to construct the latent space;
2) CNN for encoding or RNN for decoding SMILES;
2) BO on the latent space for exploring better molecules
- ChemVAE: Low valid
- GrammarVAE: 1) CFG for parsing SMILES; 2) Higher valid than ChemVAE
- SD-VAE: 1) Offline SDT check for SMILES generation; 2) Higher valid than GrammarVAE
- AAE
  - DruGAN: Applied AAE on fingerprints
  - Blaschke *et al.*: 1) AAE is more suitable for SMILES inputs than VAE; 2) Uniform distribution is better than Gaussian distribution on AAE
  - ECAAE: 1) Improved conditional generation for AAE; 2) A semisupervised extension for unknown labels; 3) A selective inhibitor of JAK3

https://doi.org/10.4155/fmc-2018-0358