New ML in Chemistry

Aaron Hart
(aaron.hart@gmail.com)

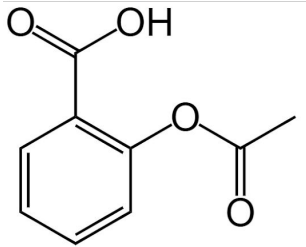
Traditional representations

Line notation
(eg. smiles, but also smarts, inchi, etc.)

O=C(C)Oc1ccccc1C(=O)O

good for text

Graph Based
(eg. sdf)



good for people

Fingerprint
(eg. morgan, k=32 and many others)

00110010110001111010110110110111

good for math

What's about the chemical hashed fingerprint?

Pros:

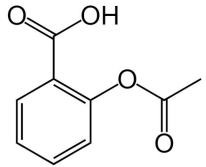
Similar molecules have similar fingerprints.

Works well for search and QSAR.

Cons:

Risk of bit collisions or sparse - pick one.

Either way the graph is lost.



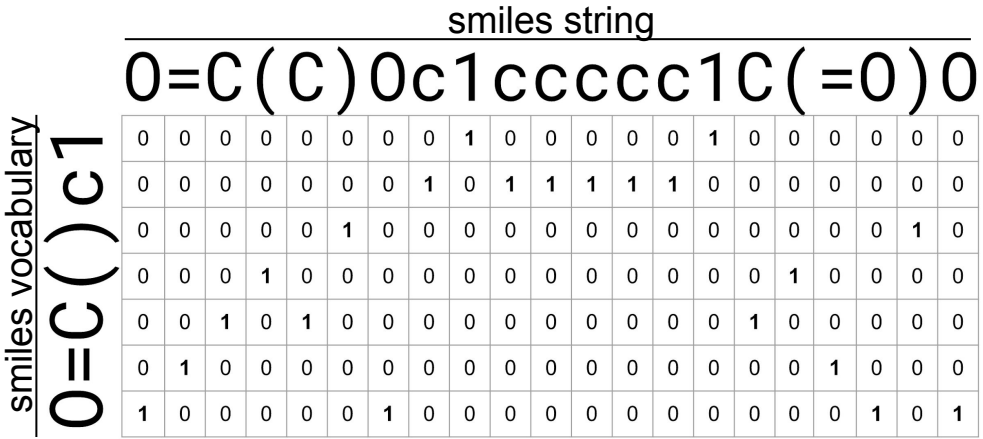
00110010110001111010110110110111

???

new representation: one hot smiles encoding.

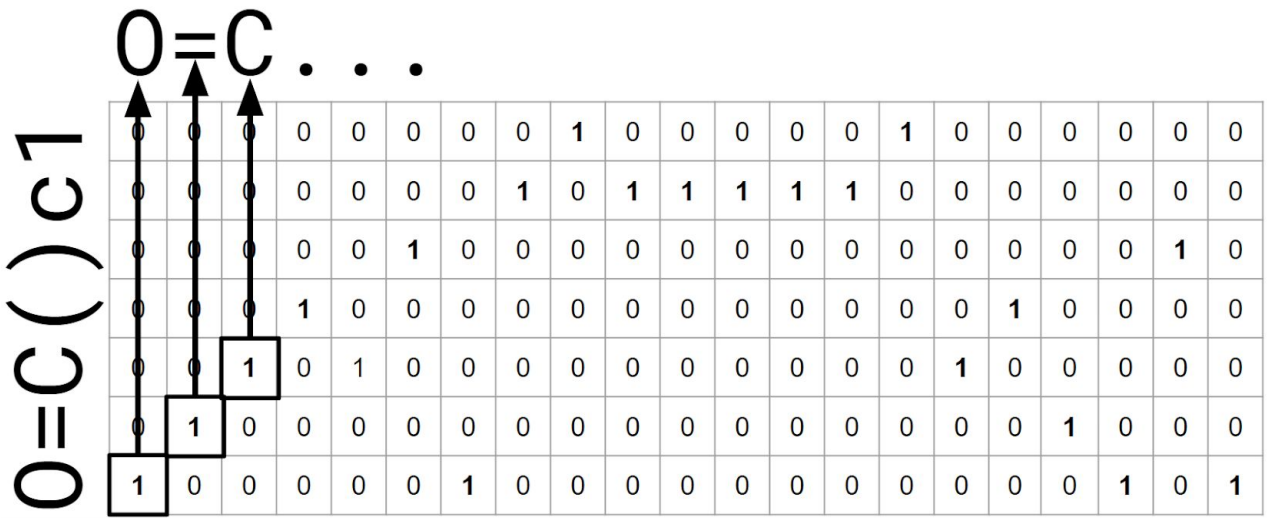
Let's steal a trick from text mining - one hot encoding, and apply it to smiles notation.

1. Scan all smiles strings to find unique ASCIIs (the vocabulary).
2. Replace each character with vector marked at vocabulary index.



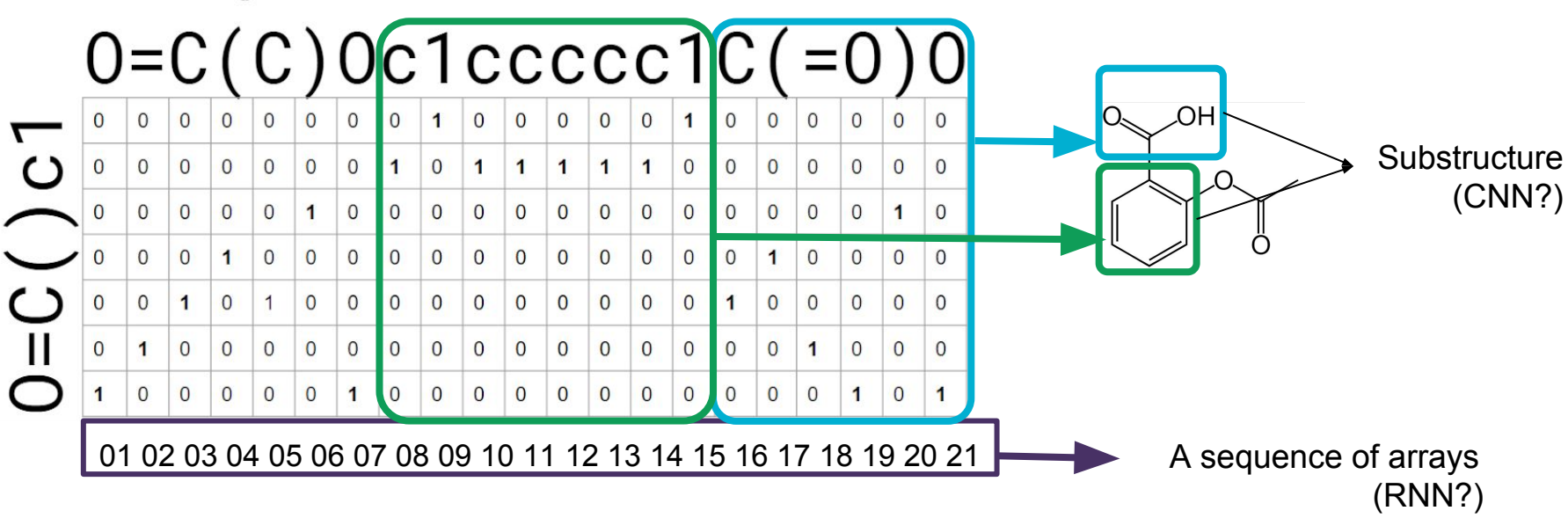
One hot smiles encoding

This is easily reversible.



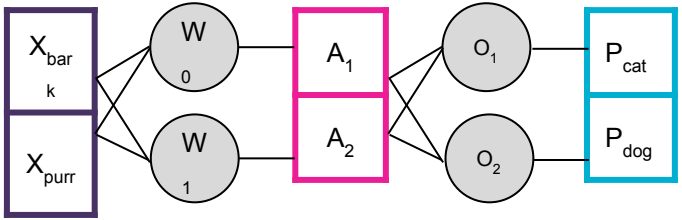
One hot smiles encoding

Smiles notation + one hot encoding has interesting implications for machine learning.



What is an artificial neural network?

a type of composite function that can express many types of real world relationships.



$\tanh(X*W+b_w) = A, \quad \text{softmax}(A*O+b_o) = P$

X_{bark}	X_{purr}
1	0
0	1
1	0

→

A_1	A_2
...	...
...	...
...	...

→

P_{cat}	P_{dog}
0	1
1	0
1	1

-

Y_{cat}	Y_{dog}
0	1
1	0
0	1

= 0?

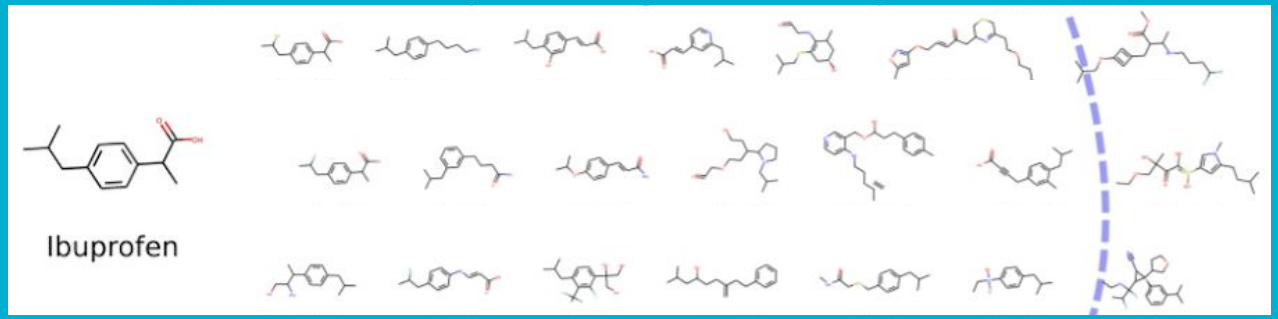
What else can we try?

O=C(C)Oc1ccccc1C(=O)O

O=C()c1

0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	**1**	0	0	0	0	0	0
0	0	0	0	0	0	0	**1**	0	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0
0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0
0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0
0	0	**1**	0	**1**	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0
0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0
1	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	**1**	0	**1**

novel compound generation with chem-vae



What is an autoencoder?

a neural network that learns to predict itself.
#wow

Conceit: forcing data through a bottleneck (latent space) requires us to learn something important about the data.

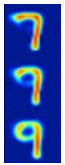
Consists of 2 components:

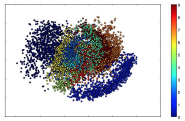
Encoder - embed input to latent space

Decoder - reconstruct input from latent space

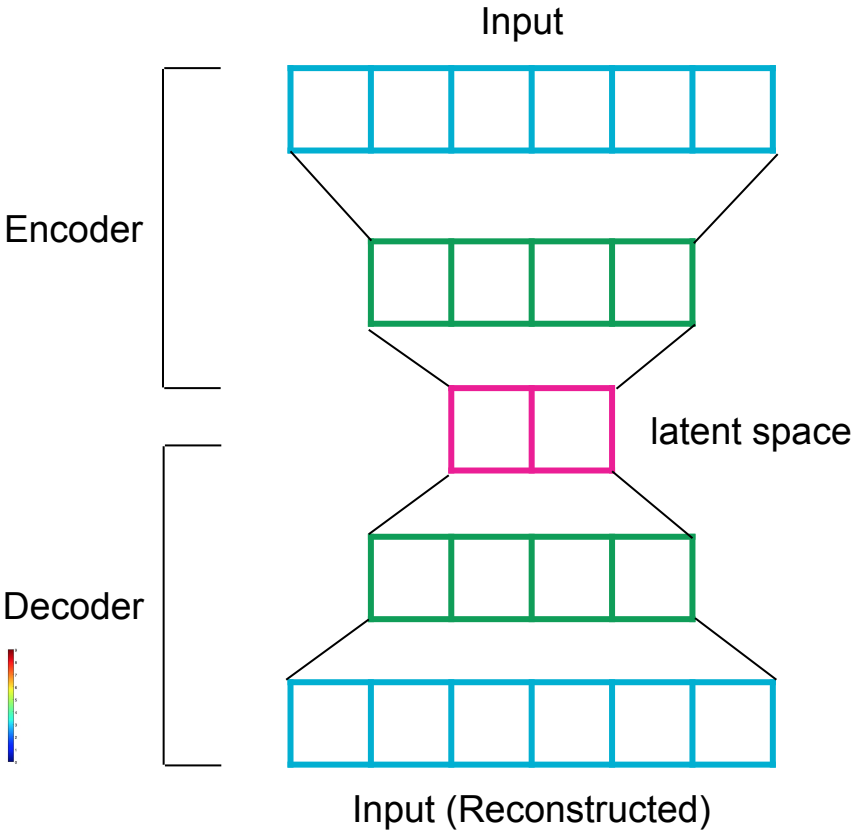
Applications:

Denoising 

Interpolation 

Visualization 

(https://blog.keras.io/building-autoencoders-in-keras.html)



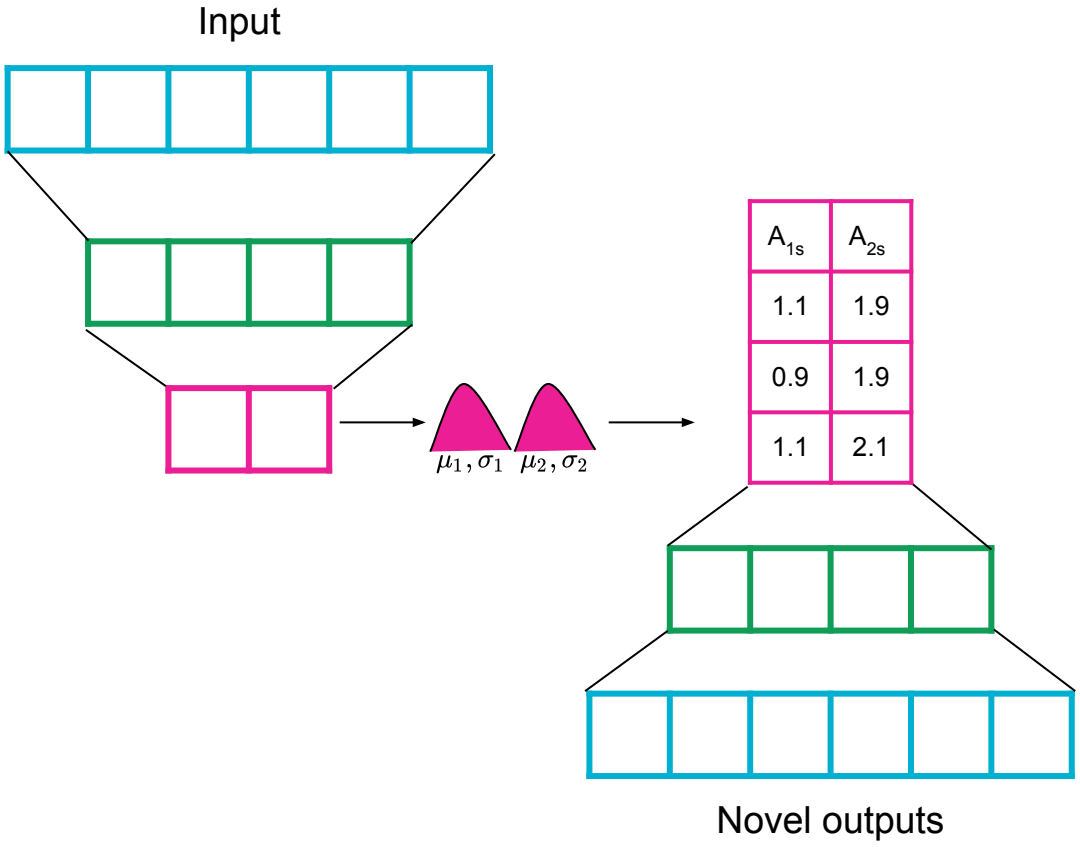
What is a variational autoencoder?

Train as usual except...

Force each scalar in latent dimension to be normally distributed.

Sample the normally distributed latent space

Use decoder to generate new outputs based on the sampled points.

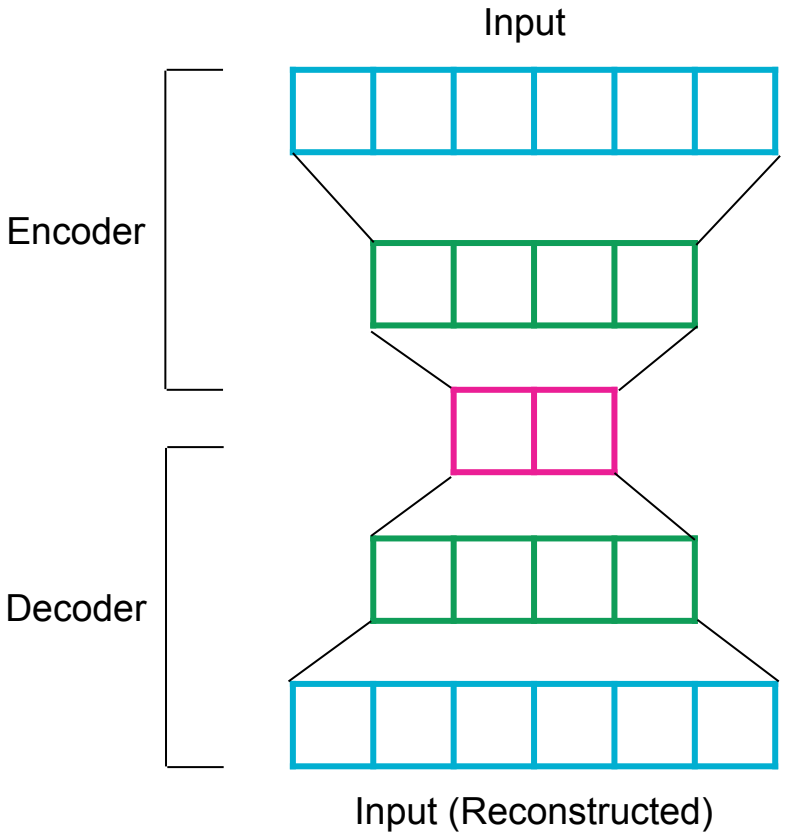


additional details

Encoder and Decoder can have wildly different architectures

Sophisticated loss functions can be used to shape the reconstructed data.

Activations at the output layer can be repeatedly sampled to generate multiple results.



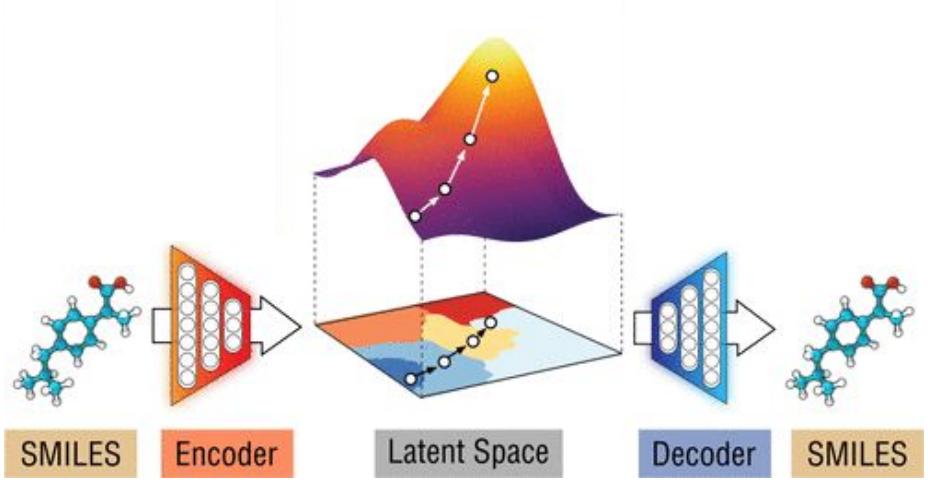
chem-vae

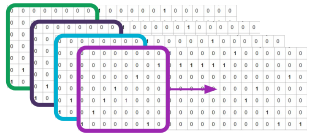
Convolutional layers used in encoder. Similar to model components used in image recognition (state of the art?)

Dense layer is vector(n=300), each element normally distributed.

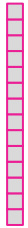
Recurrent layers used to reconstruct the initial one hot smiles sequence. (Also used in machine translation)

Incorporate property prediction in loss function to improve molecule generation.
(5*QED + SAS)

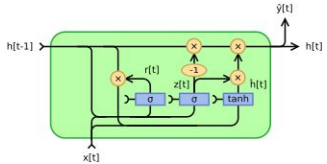




Stacked 1D CNN*

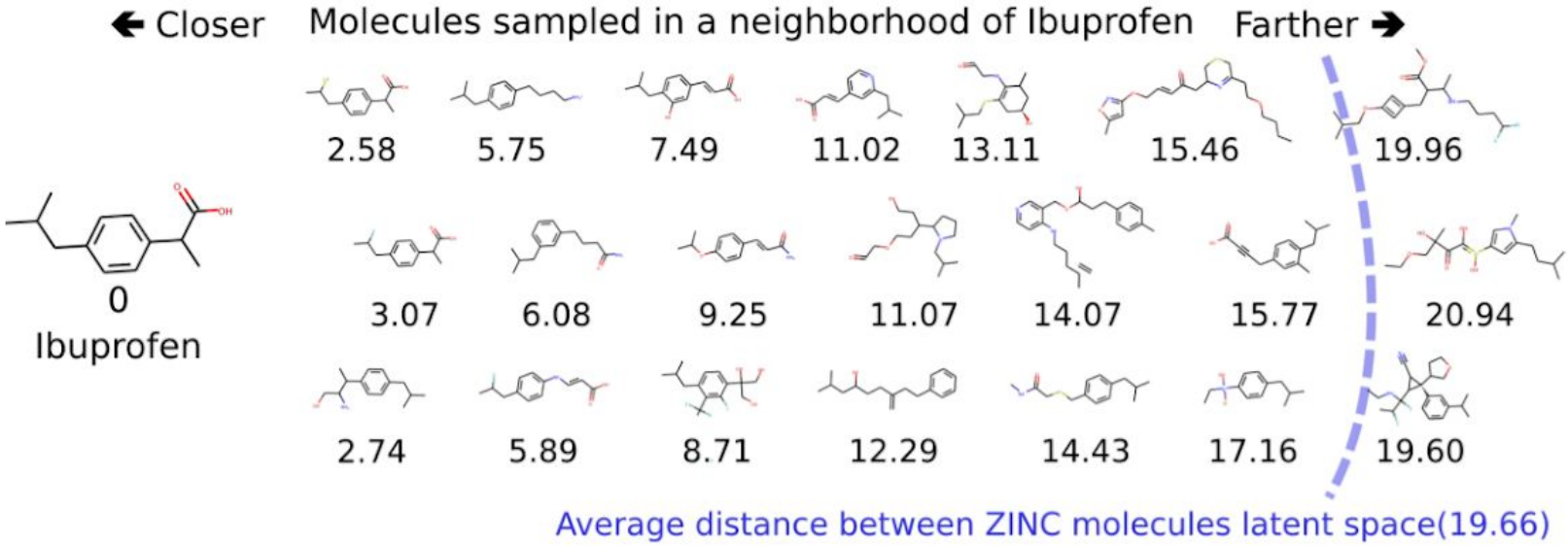


Dense

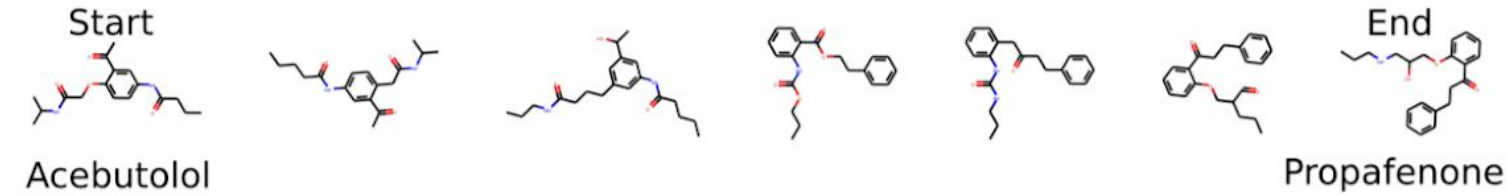


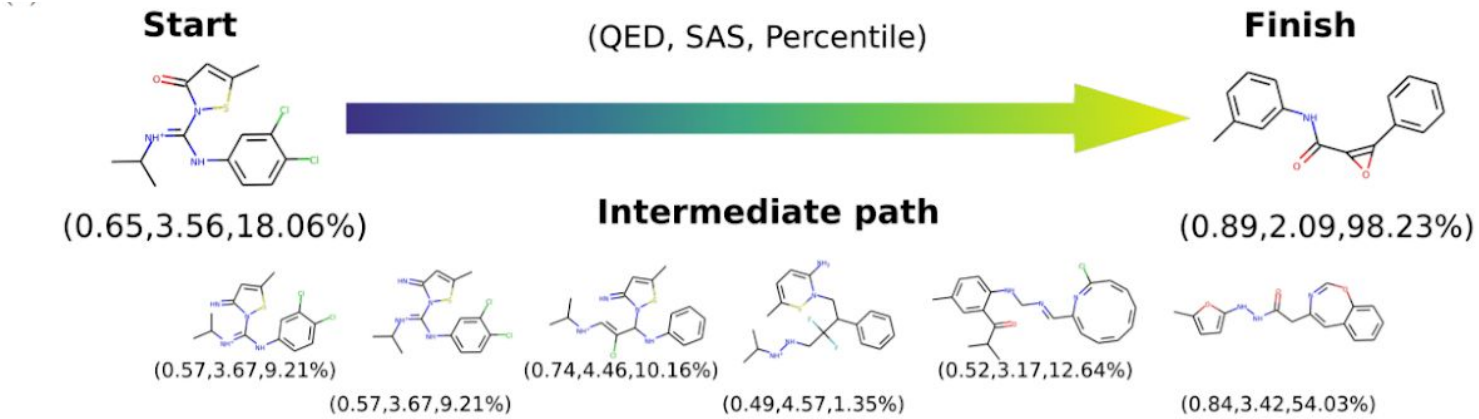
Stacked GRU

chemvae: sampling a structure



chemvae: interpolation





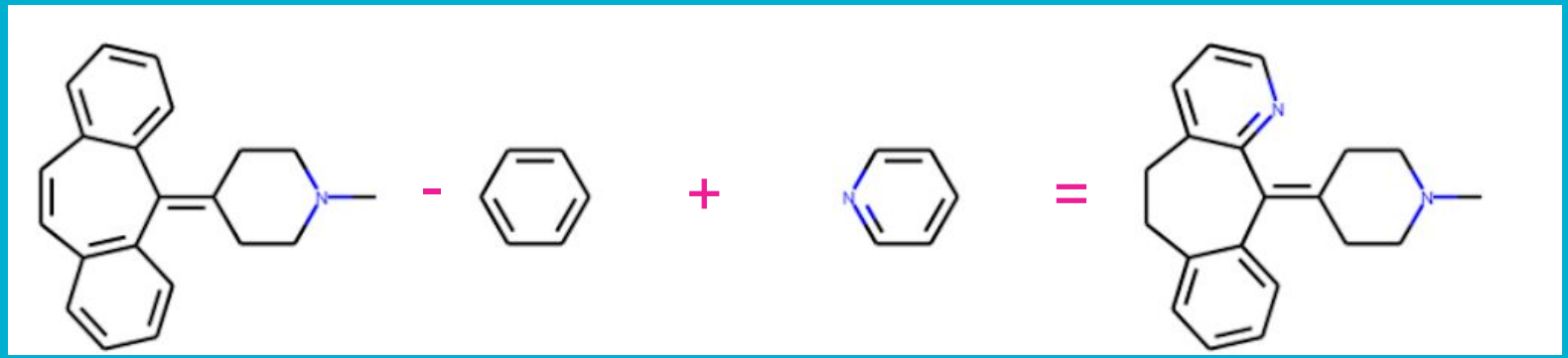
chemvae: key takeaways

Encoding into latent continuous space not a new idea.

Decoding a latent space using an RNN is potentially exciting.

Degree of real novelty is unkown

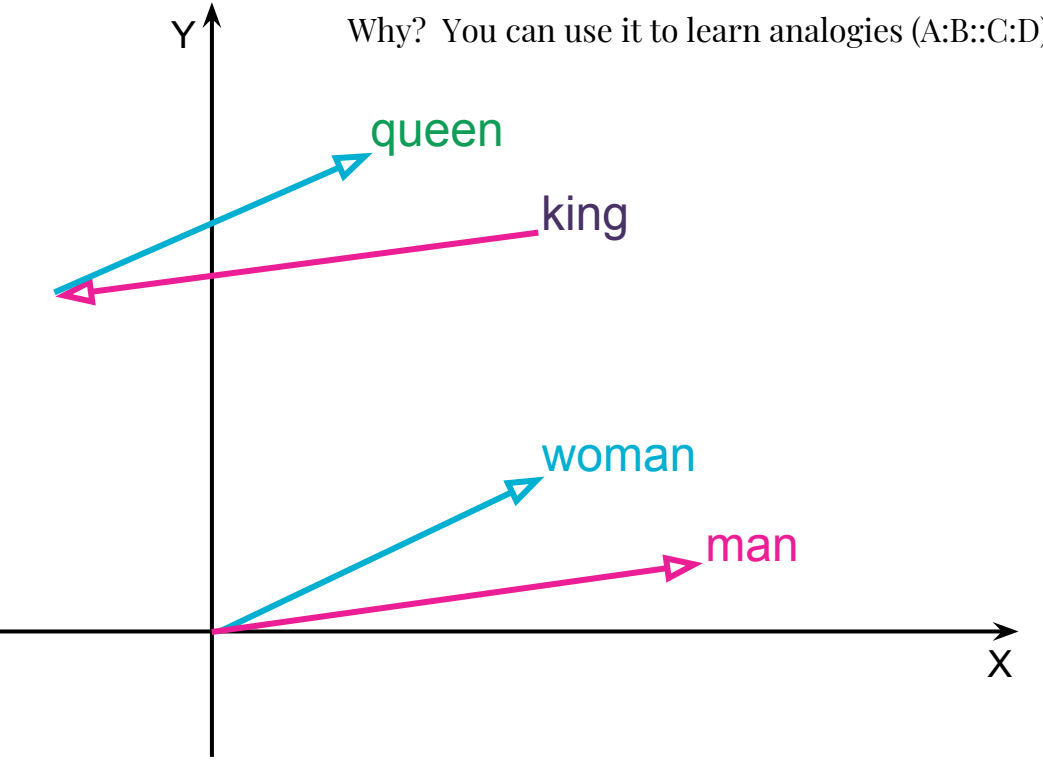
scaffold modification with mol2vec



algorithm: word2vec

*map a **word** to a **vector** representing its context.*

Why? You can use it to learn analogies (A:B::C:D)



man:king::woman:queen

vector math:
queen = **king-man+woman**

Food for thought: Could you describe a sentence as the sum of its word vectors?

word2vec: data formulation

Vocabulary = **the** quick brown fox jumps over lazy dog

The quick brown fox jumps over the lazy dog.

___ **the** quick

The **quick** brown

quick **brown** fox

brown **fox** jumps

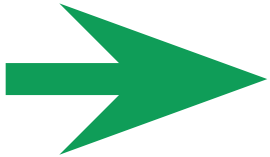
fox **jumps** over

jumps **over** the

over **the** lazy

the **lazy** dog

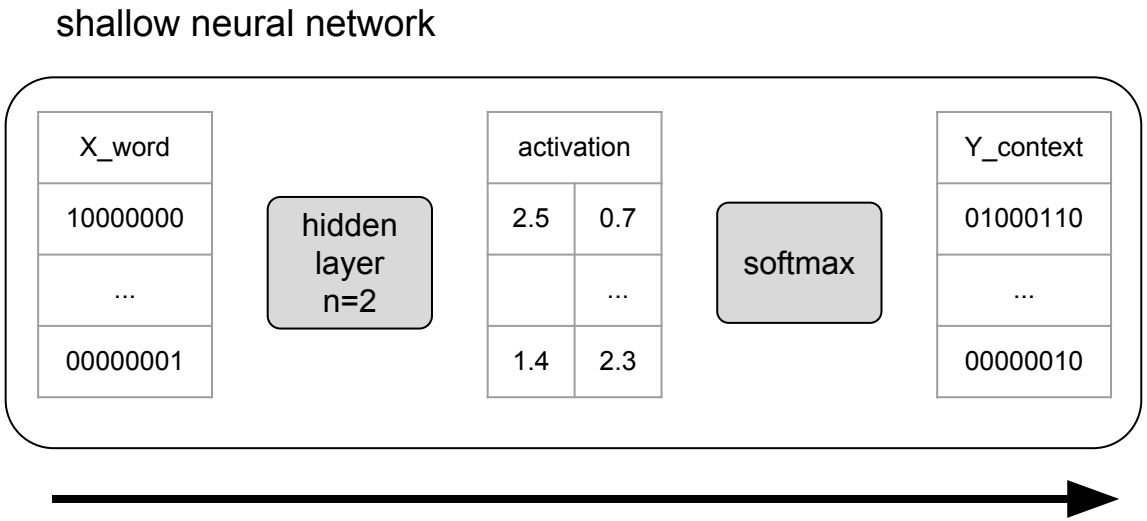
lazy **dog** ___



	X_word	Y_context
the	**1**0000000	**0**1000110
quick	01000000	10100000
brown	00100000	01010000
fox	00010000	00101000
jumps	00001000	00010100
over	00000100	00001010
lazy	00000010	00000101
dog	00000001	00000010

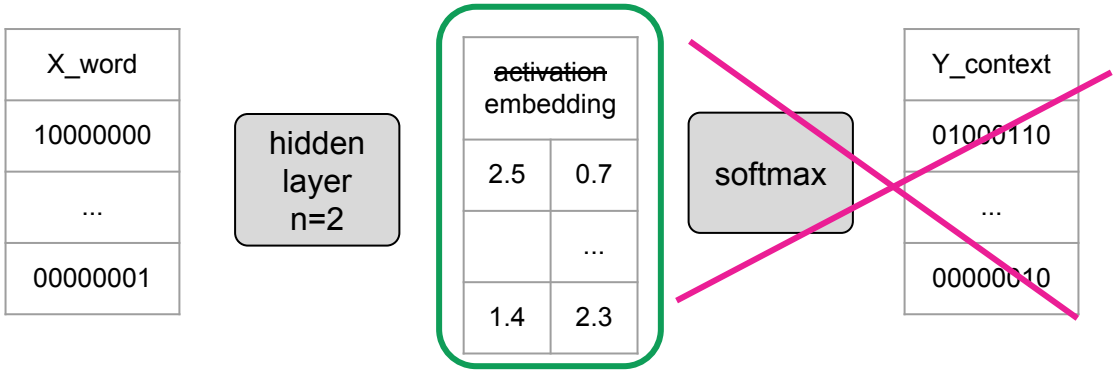
word2vec: learning an embedding

	X_word
the	10000000
quick	01000000
brown	00100000
fox	00010000
jumps	00001000
over	00000100
lazy	00000010
dog	00000001



Y_context
01000110
10100000
01010000
00101000
00010100
00001010
00000101
00000010

word2vec: extracting the embedding



	v_1	v_2
the	2.5	0.7
quick	4.5	4.8
brown	2.1	3.6
fox	5.5	1.2
jumps	3.4	4.5
over	2.7	3.7
lazy	5.0	7.3
dog	1.4	2.3
Sentence vector	**27.1**	**28.1**

https://github.com/samoturk/mol2vec

text mining:word2vec :: cheminformatics : mol2vec

Repeat everything we just saw but using *substructure* in place of *word*.

Why not call it *substructure2vec*?

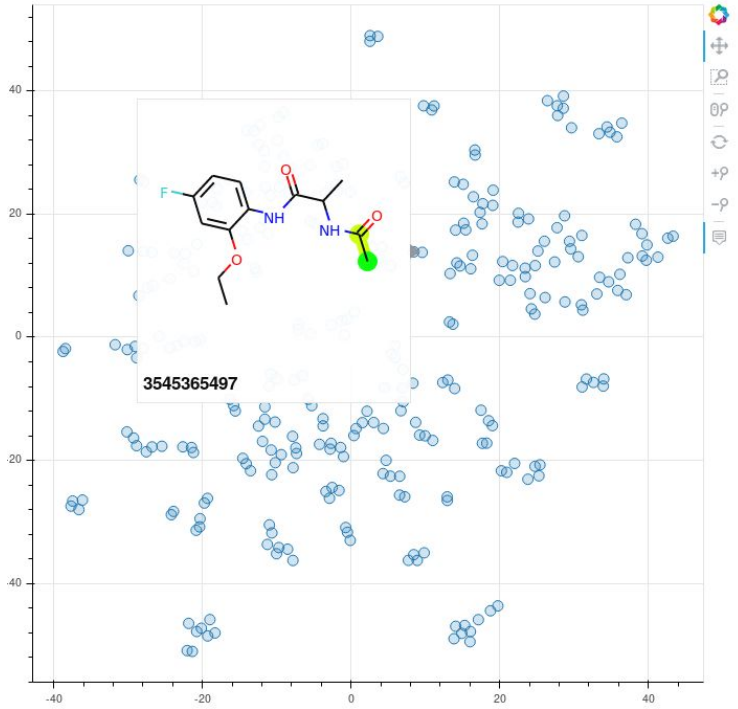
- We can get the molecular vector by summing it's substructure vectors.
- We care about molecules not substructures
- It's not a good name

Train on all of ZINC (20 million compounds)

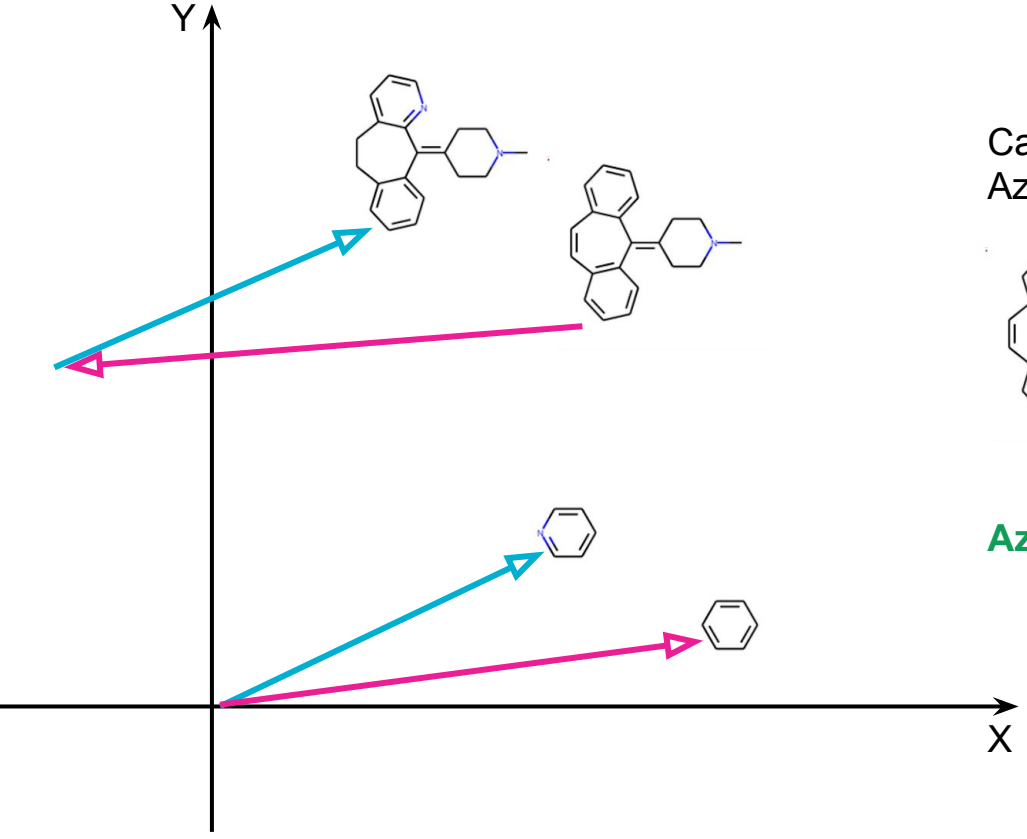
Use morgan fingerprints, radius 1 to create "chemical words"

Train neural network with 300 neurons in hidden layer.

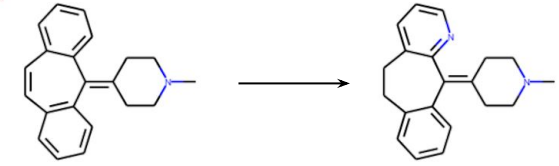
Can be trained in less than a day and inference can be done in real time.



mol2vec : scaffold modification



Can we scaffold hop from Cyproheptadine to Azatadine using vector math?

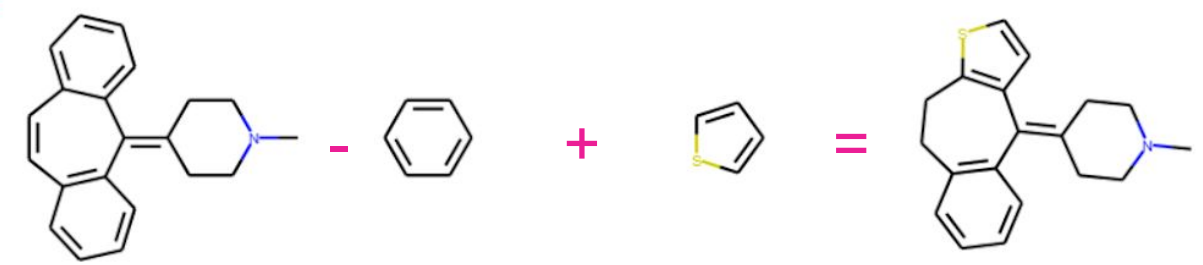


Azatadine = Cyproheptadine - Benzene + Pyridine

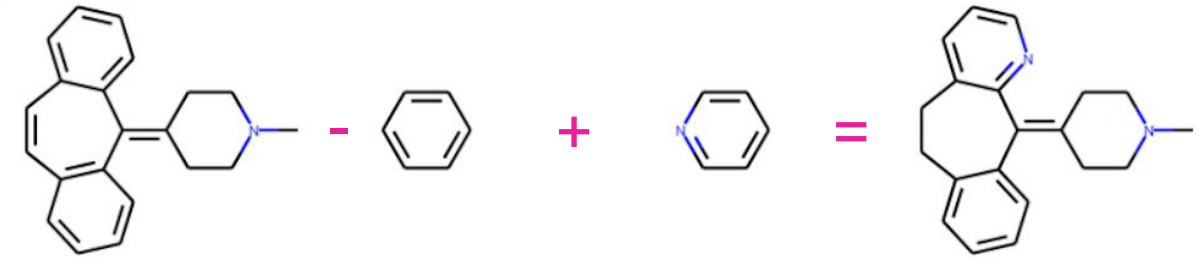
mol2vec : scaffold modification

```
Cyproheptadine = Chem.MolFromSmiles("c43\C(=C1/CCN(C)CC1)c2ccccc2\C=C/c3cccc4")
Pizotifen = Chem.MolFromSmiles("s1c3c(cc1)C(\c2c(cccc2)CC3)=C4/CCN(C)CC4")
Azatadine = Chem.MolFromSmiles("n4c3\C(=C1/CCN(C)CC1)c2ccccc2CCc3ccc4")
Benzene = Chem.MolFromSmiles("c1ccccc1")
Thiophene = Chem.MolFromSmiles("c1ccsc1")
Pyridine = Chem.MolFromSmiles("c1ccncc1")
```

```
q1 = vec(Cyproheptadine) - vec(Benzene) + vec(Thiophene)
qr1 = search(q1, library_vectors)
result = library["mols"][qr1[0]]
Draw.MolsToGridImage([Cyproheptadine, Benzene, Thiophene, result], molsPerRow=4, useSVG=False)
```



```
q2 = vec(Cyproheptadine) - vec(Benzene) + vec(Pyridine)
qr2 = search(q2, library_vectors)
result = library["mols"][qr2[0]]
Draw.MolsToGridImage([Cyproheptadine, Benzene, Pyridine, result], molsPerRow=4, useSVG=False)
```



mol2vec : other potential uses

Dimensionality reduction.
- Train on all the molecules (ahead of time)
- Infer from model for real time projection
  - Faster than t-SNE (for the user)
  - More interesting than PCA?

Chemical Semantics
- \- vec(PAINS)?
- \+ vec(Solubility)?

Supervised learning, should be no worse than Morgan FP
- property prediction
- virtual screening

References

Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules
arXiv:1610.02415

mol2vec: Unsupervised Machine Learning Approach with Chemical Intuition
10.1021/acs.jcim.7b00616

Quantitative estimate of drug likeness
10.1038/nchem.1243

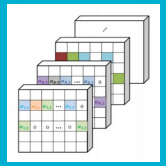
Synthetic accessibility score
10.1186/1758-2946-1-8

Classification of scaffold-hopping approaches
10.1016/j.drudis.2011.10.024

The end

B Sides

Relational graph convolution
arXiv:1802.04944



Retrosynthesis with seq2seq deep learning
10.1021/acscentsci.7b00303