Low Data Drug Discovery with One-Shot Learning

Aleksey Morozov

Moscow Institute of Physics and Technology

November 9, 2017

Plan

- Low Data Drug Discovery with One-Shot Learning
- Machine-learning approaches in drug discovery: methods and applications

Low Data Drug Discovery with One-Shot Learning Motivation

- Deep neural networks in particular have been demonstrated to provide significant boosts in predictive power when inferring the properties and activities of small-molecule compounds.
- The applicability of these techniques has been limited by the requirement for large amounts of training data.

Motivation

- One-shot learning can be used to significantly lower the amounts of data required to make meaningful predictions in drug discovery applications.
- Iterative refinement long short-term memory, when combined with graph convolutional neural networks, significantly improves learning of meaningful distance metrics over small-molecules.

Problem

- The lead optimization step of drug discovery is fundamentally a low-data problem.
- To optimize the candidate molecule by finding analogue molecules with increased pharmaceutical activity and reduced risks to the patient.

Mathematical Formalism

We consider the situation in which one has multiple binary learning tasks.

Goal: to harness the information available in the training tasks to create strong classifiers for the test systems.

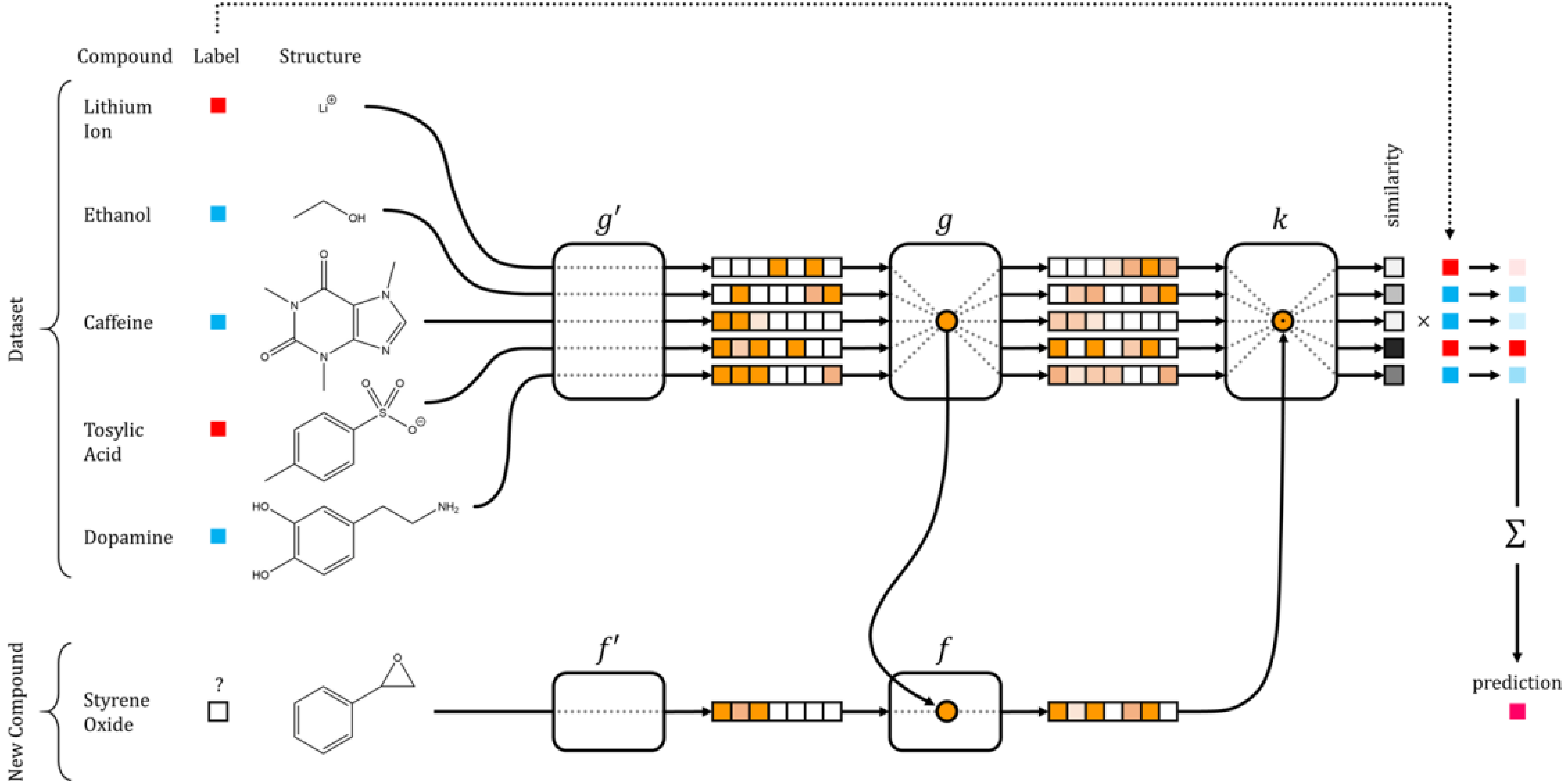
We have *T* tasks, each associated with data set

$$S = \{(x_i, y_i)\}_{i=1}^{m}, y_i \in \{0,1\}.$$

The goal is to learn a function h, parametrized upon choice of support S that predicts the probability of any query x being active in the same system.

$$h_S(x){:}\ \chi \rightarrow [0,1]$$

where χ is the chemical space of small-molecules.



Schematic of Network Architecture for one-shot learning in drug discovery

From arXiv preprint arXiv:1611.03199

One-shot learning

Let $a(x, x_i)$ denote some weighting function for query example x and support set element x_i with associated label y_i.

Then the label $h_S(x)$ for query compound x can be defined as $h_S(x) = \sum_{i=1}^{m} a(x, x_i) y_i$.

$a(x, x_i)$ - non-negative function, $\sum_{i=1}^{m} a(x, x_i) = 1$, is called attention mechanism.

$$a(x, x_i) = \frac{k\big(f(x), g(x_i)\big)}{\sum_{j=1}^{m} k\left(f(x), g\left(x_j\right)\right)}$$

$$f\colon \chi \to \mathbb{R}^p$$

$$g\colon \chi \to \mathbb{R}^p$$

f and g are graph-convolutional networks, k could be the cosine-distance.

One-shot learning

Feature map $f(x)$ is formed without any context about data available in support S.

Develop full context embeddings, in which embeddings $f(x) = f(x|S)$ and $g(x_i) = g(x_i|S)$ are computed using both x and S. Full context embeddings allow the embeddings for x and x_i to affect one another.

$$g(x|S) = \mathrm{BiLSTM}([g'(x_1)| \dots |g'(x_m)])$$

$$f(x|S) = \mathrm{attLSTM}(f'(x), \{g(x_i|S)\})$$

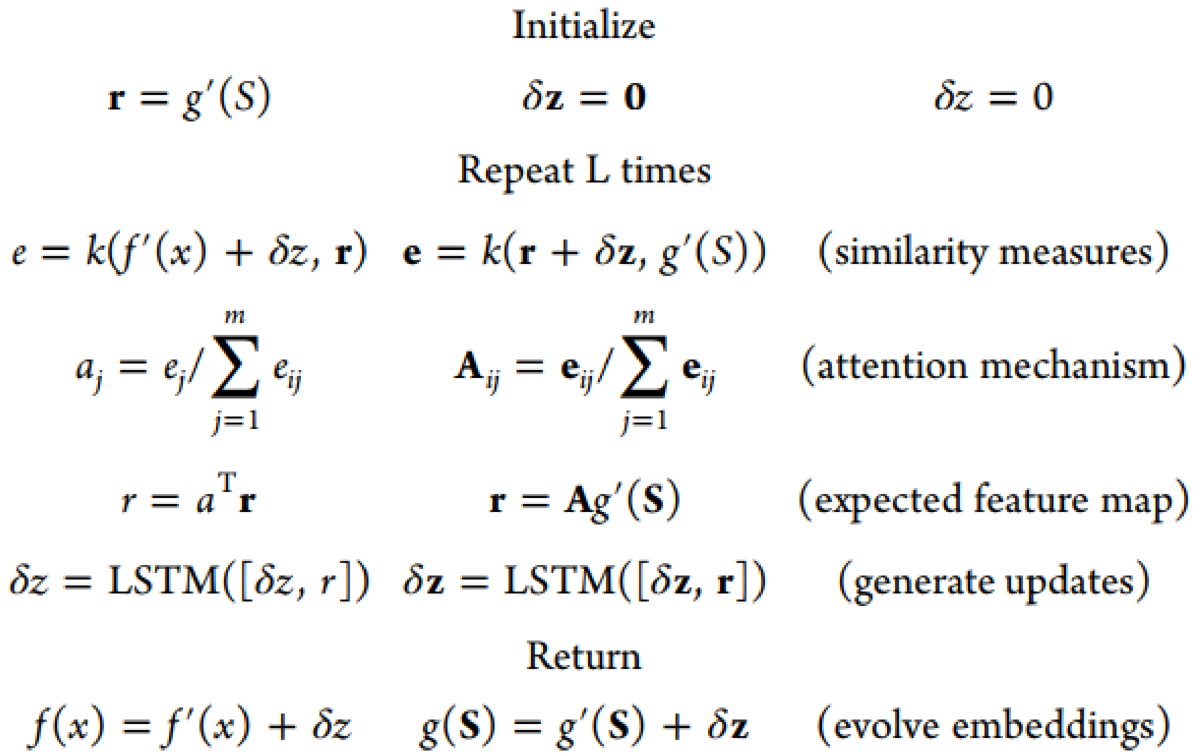
But. attLSTM is order-independent, while the BiLSTM is order dependent. Furthermore, the treatment of f and g is nonsymmetric.

Iterative Refinement LSTMs

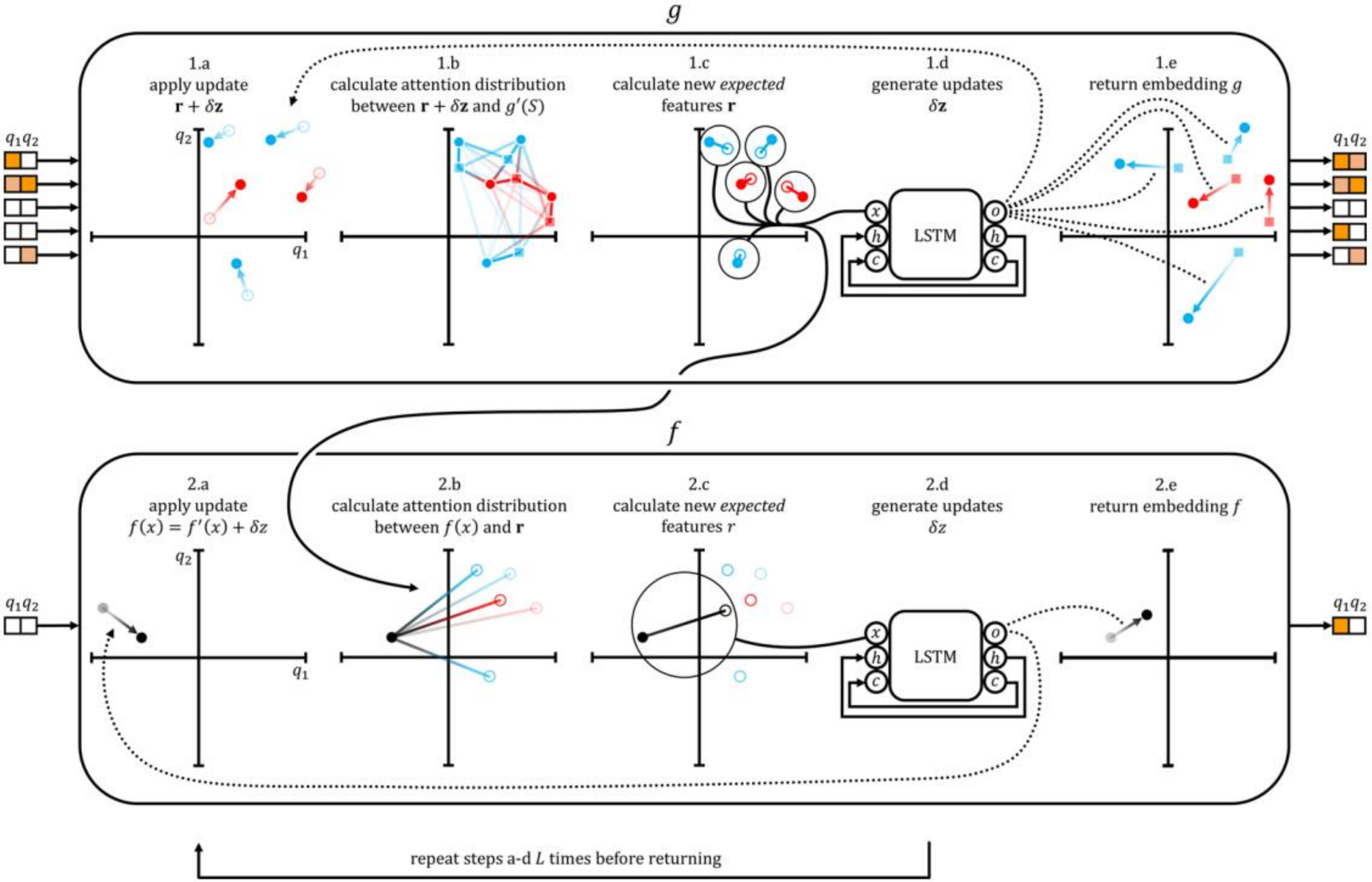
Idea: generate both query embedding f and support embedding g.

Solution: iteratively evolve both embeddings simultaneously.

- Define $f(x) = f'(x)$, $g(S) = g'(S)$.
- Iteratively update the embeddings f and g for L steps using an attention mechanism.



	Initialize	
$\mathbf{r} = g'(S)$	$\delta\mathbf{z} = \mathbf{0}$	$\delta z = 0$
	Repeat L times	
$e = k(f'(x) + \delta z, \mathbf{r})$	$\mathbf{e} = k(\mathbf{r} + \delta\mathbf{z}, g'(S))$	(similarity measures)
$a_j = e_j / \sum_{j=1}^{m} e_{ij}$	$\mathbf{A}_{ij} = \mathbf{e}_{ij} / \sum_{j=1}^{m} \mathbf{e}_{ij}$	(attention mechanism)
$r = a^{\mathrm{T}}\mathbf{r}$	$\mathbf{r} = \mathbf{A}g'(\mathbf{S})$	(expected feature map)
$\delta z = \mathrm{LSTM}([\delta z, r])$	$\delta\mathbf{z} = \mathrm{LSTM}([\delta\mathbf{z}, \mathbf{r}])$	(generate updates)
	Return	
$f(x) = f'(x) + \delta z$	$g(\mathbf{S}) = g'(\mathbf{S}) + \delta\mathbf{z}$	(evolve embeddings)



From arXiv preprint arXiv:1611.03199

Graph Convolutions

Use graph-convolutional neural networks to encode small-molecules in a form suitable for one-shot prediction.

New layers: pair of graph convolutional layer types, max-pooling and graph-gathering.

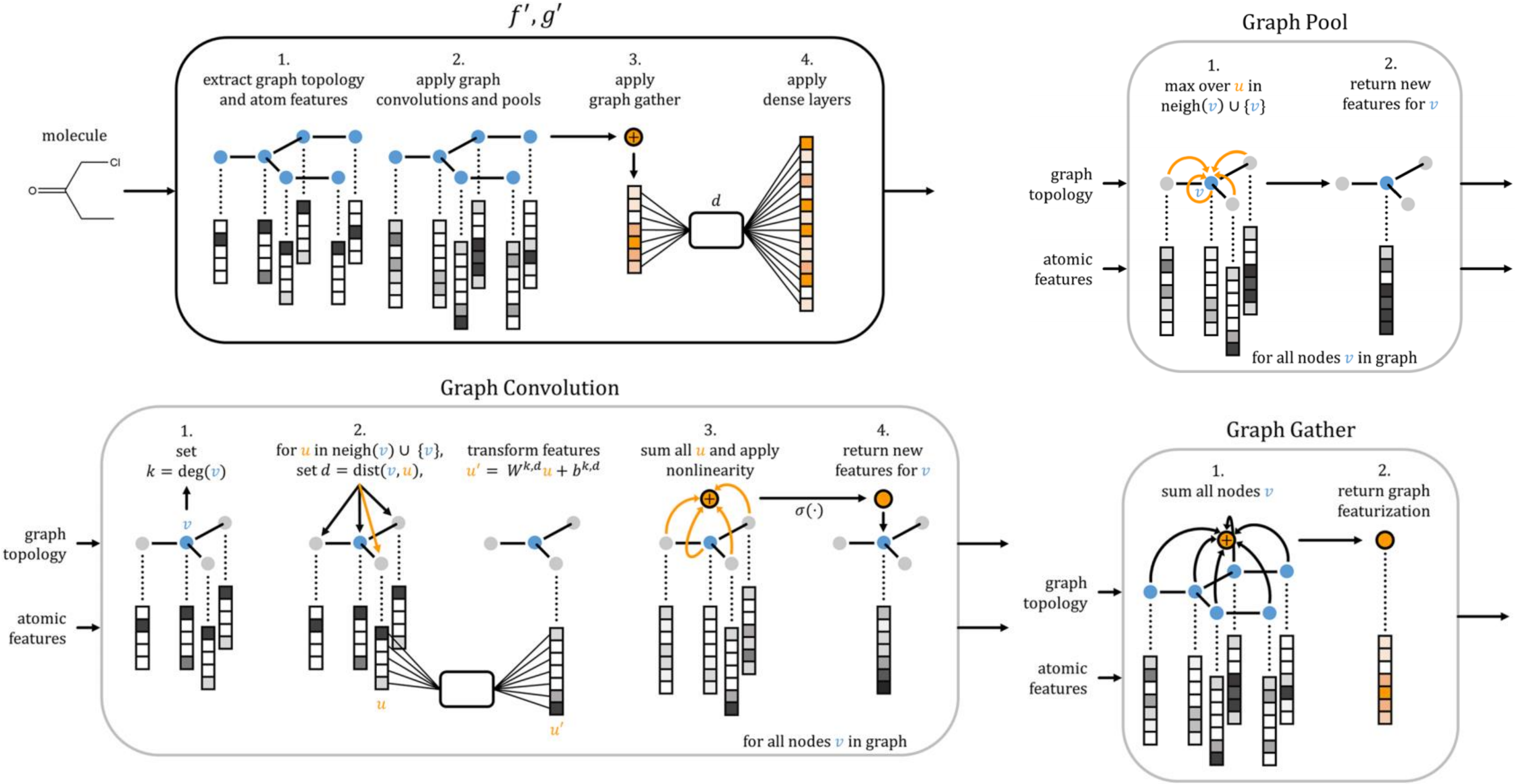
Three major neural-network layers that are used to featurizing the molecular graphs:

- the graph convolution $h_{\mathrm{conv}}(G) = [h_{\mathrm{conv}}(v_1), h_{\mathrm{conv}}(v_2), \dots];$
- the graph pool $h_{\mathrm{pool}}(G);$
- the graph gather $h_{gather}(G).$

$$h_{\mathrm{conv}}(v) = \sigma\Big(\sum_{(u,v)\in E} W^{\deg(v)} v + U^{\deg(v)} u + b^{\deg(v)} \Big)$$

$$h_{\mathrm{pool}}(v) = \max\{ \max_{(u,v)\in E} u,\ v \}$$

$$h_{\mathrm{gather}}(G) = \sum_{u\in V} u$$


f', g'
1. extract graph topology and atom features
2. apply graph convolutions and pools
3. apply graph gather
4. apply dense layers
molecule
Cl
O
d
Graph Pool
1. max over u in neigh(v) ∪ {v}
2. return new features for v
graph topology
atomic features
v
for all nodes v in graph
Graph Convolution
1. set $k = \deg(v)$
2. for u in neigh(v) ∪ {v}, set $d = \text{dist}(v, u)$,
transform features $u' = W^{k,d}u + b^{k,d}$
3. sum all u and apply nonlinearity
4. return new features for v
graph topology
atomic features
v
u
u'
$\sigma(\cdot)$
for all nodes v in graph
Graph Gather
1. sum all nodes v
2. return graph featurization
graph topology
atomic features

From arXiv preprint arXiv:1611.03199

Model Training and Evaluation

Tasks represent the set of all learning tasks.

Split this into two disjoint sets, Train−Tasks and Test−Tasks. Let S represent a support set, and let B represent a batch of queries.

$$\mathcal{L} = -E_{T \in Train-Tasks}\left[E_{S \sim T, B \sim T}\left[\sum_{(x,y) \in B} \log P_\theta(y|x,S)\right]\right]$$

Task T is randomly sampled, and then a support S and a batch of queries B are sampled from the task.

One gradient descent step minimizing $\mathcal{L}$, using ADAM is performed for each episode.

One-shot models training

- An assay from the training assays is randomly sampled.
- A support S of size $n_{\text{pos}} + n_{\text{neg}}$ and a batch of queries B are sampled from the task.
- Models were trained for $2000 \cdot n_{\text{train}}$ episodes.
- Each episode takes one gradient descent step minimizing $\mathcal{L}$, using ADAM.
- At test time, the accuracy of a one-shot model is evaluated separately on each testing assay.
- For a given test assay, a support of size $n_{\text{pos}} + n_{\text{neg}}$ is sampled at random from data points for that assay.
- The ROC-AUC score for the one-shot model is evaluated on the remainder of the data points for the test assay.
- This procedure is reported 20 times for each test assay, and the mean and standard deviations of computed ROC-AUC scores for each test assay are presented in the tables.

Singletask models training

- Random forests were trained on circular fingerprint representations of input molecules.

- For each test assay, supports of size $n_{\text{pos}} + n_{\text{neg}}$ are sampled at random.

- The random forest model is trained on this sampled supported set and evaluated on the remainder of test assay.

- This procedure is repeated 20 times.

- Singletask graph convolutional networks are trained as the random forests are, but with graph convolutional features instead of circular fingerprint representations.

Experiments I

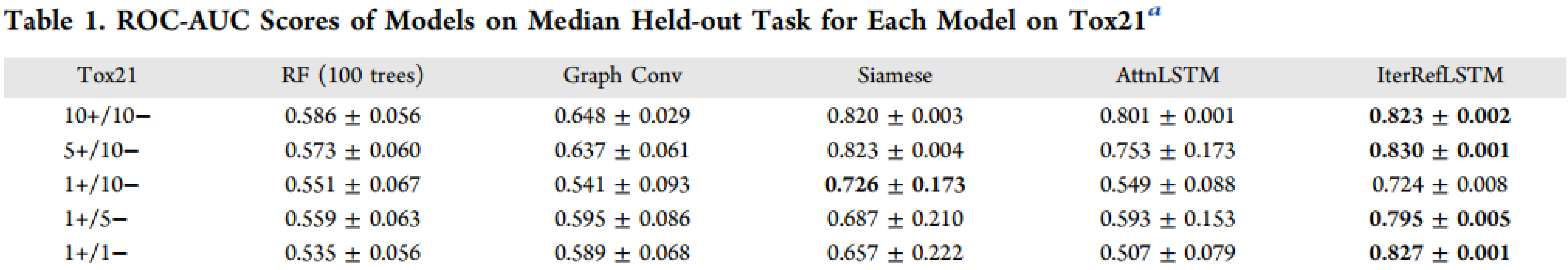


Table 1. ROC-AUC Scores of Models on Median Held-out Task for Each Model on Tox21[a]

Tox21	RF (100 trees)	Graph Conv	Siamese	AttnLSTM	IterRefLSTM
10+/10−	0.586 ± 0.056	0.648 ± 0.029	0.820 ± 0.003	0.801 ± 0.001	**0.823 ± 0.002**
5+/10−	0.573 ± 0.060	0.637 ± 0.061	0.823 ± 0.004	0.753 ± 0.173	**0.830 ± 0.001**
1+/10−	0.551 ± 0.067	0.541 ± 0.093	**0.726 ± 0.173**	0.549 ± 0.088	0.724 ± 0.008
1+/5−	0.559 ± 0.063	0.595 ± 0.086	0.687 ± 0.210	0.593 ± 0.153	**0.795 ± 0.005**
1+/1−	0.535 ± 0.056	0.589 ± 0.068	0.657 ± 0.222	0.507 ± 0.079	**0.827 ± 0.001**

The Tox21 collection consists of 12 (9+3) nuclear receptor assays related to human toxicity.

All one-shot learning methods show strong boosts over the random-forest baseline, with the iterative refinement LSTM models showing a more robust boost in the presence of less data

From arXiv preprint arXiv:1611.03199

Experiments II

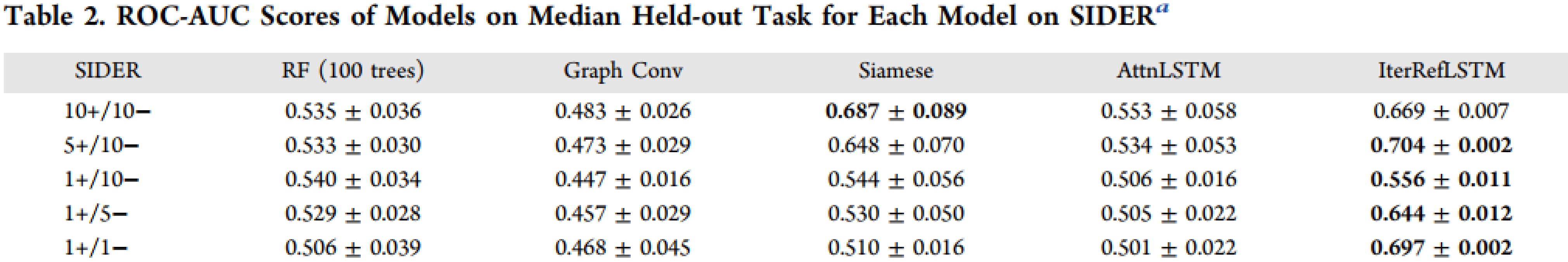


Table 2. ROC-AUC Scores of Models on Median Held-out Task for Each Model on SIDER[a]

SIDER	RF (100 trees)	Graph Conv	Siamese	AttnLSTM	IterRefLSTM
10+/10−	0.535 ± 0.036	0.483 ± 0.026	**0.687 ± 0.089**	0.553 ± 0.058	0.669 ± 0.007
5+/10−	0.533 ± 0.030	0.473 ± 0.029	0.648 ± 0.070	0.534 ± 0.053	**0.704 ± 0.002**
1+/10−	0.540 ± 0.034	0.447 ± 0.016	0.544 ± 0.056	0.506 ± 0.016	**0.556 ± 0.011**
1+/5−	0.529 ± 0.028	0.457 ± 0.029	0.530 ± 0.050	0.505 ± 0.022	**0.644 ± 0.012**
1+/1−	0.506 ± 0.039	0.468 ± 0.045	0.510 ± 0.016	0.501 ± 0.022	**0.697 ± 0.002**

The SIDER data set contains information grouped the drug−SE pairs into 27 (21+6) system organ classes according to MedDRA classifications.

The Siamese and IterRefLSTM methods show strong boosts over the random-forest baseline, but the AttnLSTM has poor performance on this collection (comparable to the random forest). The iterative refinement LSTM models show a robust boost in the presence of less data. The graph convolutional baseline does poorly, with performance indistinguishable from random.

From arXiv preprint arXiv:1611.03199

Experiments III

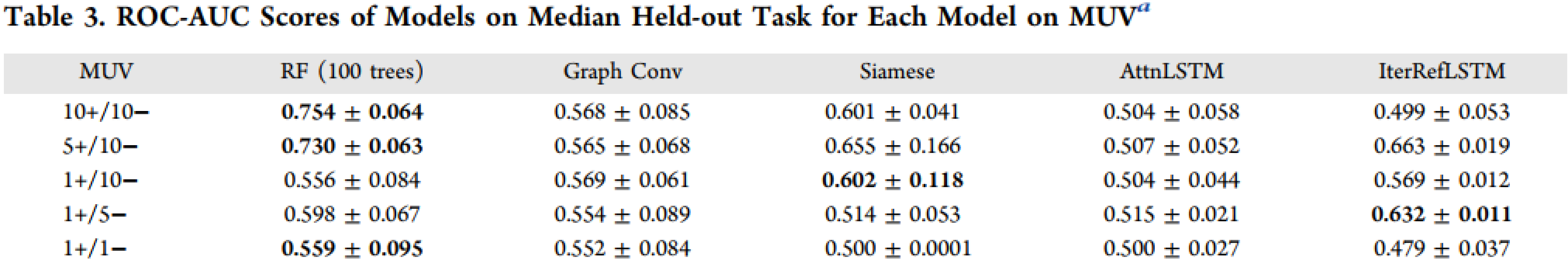


Table 3. ROC-AUC Scores of Models on Median Held-out Task for Each Model on MUV[a]

MUV	RF (100 trees)	Graph Conv	Siamese	AttnLSTM	IterRefLSTM
10+/10−	**0.754 ± 0.064**	0.568 ± 0.085	0.601 ± 0.041	0.504 ± 0.058	0.499 ± 0.053
5+/10−	**0.730 ± 0.063**	0.565 ± 0.068	0.655 ± 0.166	0.507 ± 0.052	0.663 ± 0.019
1+/10−	0.556 ± 0.084	0.569 ± 0.061	**0.602 ± 0.118**	0.504 ± 0.044	0.569 ± 0.012
1+/5−	0.598 ± 0.067	0.554 ± 0.089	0.514 ± 0.053	0.515 ± 0.021	**0.632 ± 0.011**
1+/1−	**0.559 ± 0.095**	0.552 ± 0.084	0.500 ± 0.0001	0.500 ± 0.027	0.479 ± 0.037

The MUV data set collection contains 17 (12+5) assays designed to be challenging for standard virtual screening.

The positives examples in these data sets are selected to be structurally distinct from one another. As a result, this collection is a best-case scenario for baseline machine learning and a worst-case test for the low-data methods, since structural similarity cannot be effectively exploited to predict behavior of new active compounds. One-shot learning methods may have some difficulties generalizing to new molecular scaffolds.

From arXiv preprint arXiv:1611.03199

ML in drug discovery

- SVMs are used for binary property or activity predictions, for example, to distinguish between drugs and nondrugs or between compounds that have or do not have specific activity, synthetic accessibility, or aqueous solubility.

- The DT approach has been applied to problems such as designing combinatorial libraries, predicting ‘drug-likeness’, predicting specific biological activities, and generating some specific compound profiling data.

- Naive Bayesian classifiers are frequently used in chemoinformatics both alongside or compared against other classifiers, generally for predicting biological rather than physicochemical properties.

- The k-NN algorithm is a simple and intuitive method to predict the class, property, or rank of a molecule based on nearest training examples in the feature space.

- ANNs have been applied in compound classification, QSAR studies, primary VS of compounds, identification of potential drug targets, and localization of structural and functional features of biopolymers.

References I

- Han Altae-Tran, Bharath Ramsundar, Aneesh S. Pappu, Vijay Pande (2017)

Low Data Drug Discovery with One-shot Learning

arXiv preprint arXiv:1611.03199

- Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, Daan Wierstra (2016)

Matching Networks for One Shot Learning

arXiv preprint arXiv:1606.04080

- David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel,

Alán Aspuru-Guzik, Ryan P. Adams (2015)

Convolutional Networks on Graphs for Learning Molecular Fingerprints

arXiv preprint arXiv:1509.09292

References II

- Diederik P. Kingma, Jimmy Ba (2017)

Adam: A Method for Stochastic Optimization

arXiv preprint arXiv:1412.6980

- Diederik P. Kingma, Jimmy Ba. (2015)

Adam: A Method for Stochastic Optimization.

arXiv preprint arXiv:1412.6980

- Lavecchia A. (2014)

Machine-learning approaches in drug discovery: methods and applications.

Drug Discovery Today, Volume 20, Number 3, March 2015.