Molecular Modeling
Learning molecular representation; Property Prediction
Connection to Biology

Modeling Metabolism: Enzyme+Metabolite Interactions

Metabolites are small molecules

L-glutamate

building block of peptidoglycan
used in bacterial cell walls

D-glutamate

Glutamate racemase
Connection to Biology
Understanding Disease and Finding Therapeutic Interventions

ribose-5-phosphate (R5P)

phospho-ribosyl pyrophosphate (PRPP)

PRPP amidotransferase

5-phosphoribosylamine

Mercaptopurine (6-MP)

Tioguanine (6-TG)
Connection to Biology
Understanding Mechanism of Actions

Rifampicin bound to RNA polymerase of *E. coli*
Design Challenges

- representation
- properties
- generation
- optimization
- interpretation

...
Machine Learning Tasks

**Property prediction**

- Single molecule screening
  - from assay(s) to property predictors

- Combination screening
  - from assay(s) to molecular cocktails

**Generative modeling**

- Molecular translation
  - from a promising lead to a better drug

- De-novo inverse molecular design
  - realizing novel molecules satisfying multiple criteria
Machine Learning Tasks

**Property prediction**

- Single molecule screening
  - from assay(s) to property predictors

- Combination screening
  - from assay(s) to molecular cocktails

**Generative modeling**

- Molecular translation
  - from a promising lead to a better drug

- De-novo inverse molecular design
  - realizing novel molecules satisfying multiple criteria
# Classes of properties (and Datasets)

<table>
<thead>
<tr>
<th>data set</th>
<th>category</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QM7, QM8, QM9</td>
<td>quantum mechanics</td>
<td>computer-generated quantum mechanics</td>
</tr>
<tr>
<td>ESOL</td>
<td>physical chemistry</td>
<td>water solubility</td>
</tr>
<tr>
<td>FreeSolv</td>
<td>physical chemistry</td>
<td>hydration free energy in water</td>
</tr>
<tr>
<td>Lipophilicity</td>
<td>physical chemistry</td>
<td>octanol/water distribution coefficient</td>
</tr>
<tr>
<td>PDBbind</td>
<td>biophysics</td>
<td>protein binding affinity</td>
</tr>
<tr>
<td>PCBA</td>
<td>biophysics</td>
<td>assorted biological assays</td>
</tr>
<tr>
<td>MUV</td>
<td>biophysics</td>
<td>assorted biological assays</td>
</tr>
<tr>
<td>HIV</td>
<td>biophysics</td>
<td>inhibition of HIV replication</td>
</tr>
<tr>
<td>BACE</td>
<td>biophysics</td>
<td>inhibition of human β-secretase 1</td>
</tr>
<tr>
<td>BBBP</td>
<td>physiology</td>
<td>ability to penetrate the blood-brain</td>
</tr>
<tr>
<td>Tox21</td>
<td>physiology</td>
<td>toxicity</td>
</tr>
<tr>
<td>ToxCast</td>
<td>physiology</td>
<td>toxicity</td>
</tr>
<tr>
<td>SIDER</td>
<td>physiology</td>
<td>side effects of drugs</td>
</tr>
</tbody>
</table>

https://pubs.acs.org/doi/10.1021/acs.jcim.9b00237#
Functional Groups Drive Properties
Screening Output
Fixed Molecular Representation

molecular fingerprint
(calculated local properties)

learned property
Learned Molecular Representation

Desired Embedding Geometry

Small, soluble

Large, soluble

Large, insoluble
Molecules can be viewed as richly annotated graphs (e.g., cephalosporin).

Together, these structural features and annotations give rise to various properties (e.g., potency, toxicity, solubility, etc).
Alternative Representations of Molecules

Tyrosine

2D graph

Conformer samples

16-bit Morgan Fingerprint

| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |

SMILES

C1=CC(=CC=C1C@@H)(C=O)O)N=O
Many drugs act by binding to a protein target (e.g., inhibiting the protein function). The binding affinity depends on the specific favorable 3D “conformations” that the drug molecule can take.
Machine Learning for Property Prediction

- **Formulation:** many ways we could try to predict properties
  - predict each property individually? Jointly?
  - as regression, classification, or ranking?
  - as classification over pairs (similar activity?)
  - etc.

- **Representation:** how do we (learn to) represent molecules?
  - fingerprint or learned neural representation

- **Method:** which machine learning technique to use?
  - MLP/random forest on top of fingerprints, RNN (smiles), GNNs on graphs, transformer, etc.
Simple Fingerprint

- We can always map a complex object into a feature vector by checking which pieces it contains ("bag of words")

- In keyed fingerprints (binary vectors) each coordinate specifies whether a particular substructure is present in the molecule
Modern Fingerprints

- A bit more sophisticated fingerprints are not restricted to pre-defined vocabularies but rather hash local substructures.
- E.g., circular / extended connectivity fingerprints

[Landrum 2012]
Compounds can be vectorized in many different ways.

**Structures**

```
COC1=C(OC)C=C2C(NC3=CC(Cl)=CC=C3)=NC=NC2=C1
```

**SMILES strings**

```
COC1=C(OC)C=C2C(NC3=CC(Cl)=CC=C3)=NC=NC2=C1
```

**Images of cells treated with the compounds**

**GNN**

**CNN**

**RNN/Transformers**
Each node in the graph is an atom with features (edges or bonds may also have features)

Our goal is to derive new, inferred atom (and graph) vectors that better reflect their surroundings in the graph

We do this by passing messages between the nodes, get new vectors, and optimize the parameter for the end task
Multi-layer Perceptron (MLP)

Each “layer” computes a linear transformation of its input

\[
\begin{pmatrix}
w_1 & w_2 & w_3 & w_4 \\
w_1 & w_2 & w_3 & w_4 \\
w_1 & w_2 & w_3 & w_4 \\
w_1 & w_2 & w_3 & w_4 \\
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
\end{pmatrix} + 
\begin{pmatrix}
b \\
b \\
b \\
b \\
\end{pmatrix} = 
\begin{pmatrix}
w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 + b \\
w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 + b \\
w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 + b \\
w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 + b \\
\end{pmatrix} \rightarrow \begin{pmatrix}
a_1 \\
a_2 \\
a_3 \\
a_4 \\
\end{pmatrix}
\]
Multi-layer Perceptron (MLP)

Stacking linear layers remains linear!

\[ C(Ax + b) + d = CA \times + (Cb + d) \]

Note: this is a 4x3 transformation instead of 4x5 in the picture to the left.

Multi-layer Perceptron (MLP)

Activation functions introduce “non-linearities”, so that we can model more complex functions

Note: this is a 4x3 transformation instead of 4x5 in the picture to the left.
Multi-layer Perceptron (MLP)

Example non-linearities

\[ \sigma(z) = \frac{1}{1 + e^{-z}} \]

Sigmoid

\[ \sigma(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \]

Tanh

\[ \text{ReLU}(z) = \begin{cases} z, & z > 0 \\ 0, & \text{otherwise} \end{cases} \]

ReLU

\[ \text{LeakyReLU}(z) = \begin{cases} z, & z > 0 \\ az, & \text{otherwise} \end{cases} \]

LeakyReLU
Multi-layer Perceptron (MLP)

We need them to handle not linearly separable data!

Recall CNNs...

Encode local neighborhoods (patches)

Encode local neighborhoods (nodes that can be reached within $K$ hops)
Big picture of Graph Neural Networks (GNNs)

Input: a graph with node feature vectors
Big picture of Graph Neural Networks (GNNs)

1. Node embeddings: encode the local neighborhood of each node separately via aggregations
Big picture of Graph Neural Networks (GNNs)

1. Node embeddings:
   encode the local neighborhood of each node separately via aggregations
Big picture of Graph Neural Networks (GNNs)

1. Node embeddings: encode the local neighborhood of each node separately via aggregations
Big picture of Graph Neural Networks (GNNs)

1. **Node embeddings**: encode the local neighborhood of each node separately via aggregations.
Big picture of GNNs

1. **Node embeddings:** encode the local neighborhood of each node separately via aggregations.

2. **Readout:** encode the entire graph by aggregating node embeddings.
Big picture of GNNs

1. **Node embeddings**: encode the local neighborhood of each node separately via aggregations

2. **Readout**: encode the entire graph by aggregating node embeddings
Encoding neighborhoods: message passing

Encoding neighborhoods: message passing

Encoding neighborhoods: message passing

Encoding neighborhoods: message passing

Encoding neighborhoods: general form

(Battaglia et al., 2016; Defferrard et al., 2016; Duvenaud et al., 2015; Hamilton et al., 2017a; Kearnes et al., 2016; Kipf & Welling, 2017; Li et al., 2016; Velickovic et al., 2018; Verma & Zhang, 2018; Ying et al., 2018b; Zhang et al., 2018)
Encoding neighborhoods: general form

In each round $k$:

**Aggregate** over neighbors

$$m_{\mathcal{N}(v)}^{(k)} = \text{AGGREGATE}^{(k)}\left(\{h_u^{(k-1)} : u \in \mathcal{N}(v)\}\right)$$

**Update: Combine** with current node

$$h_v^{(k)} = \text{COMBINE}^{(k)}\left(h_v^{(k-1)}, m^{(k)}_{\mathcal{N}(v)}\right)$$

(Battaglia et al., 2016; Defferrard et al., 2016; Duvenaud et al., 2015; Hamilton et al., 2017a; Kearnes et al., 2016; Kipf & Welling, 2017; Li et al., 2016; Velickovic et al., 2018; Verma & Zhang, 2018; Ying et al., 2018b; Zhang et al., 2018)
In each round $k$:

**Aggregate** over neighbors

$$m_{\mathcal{N}(v)}^{(k)} = \text{AGGREGATE}^{(k)}\left(\{h_u^{(k-1)} : u \in \mathcal{N}(v)\}\right)$$

**Update: Combine** with current node

$$h_v^{(k)} = \text{COMBINE}^{(k)}\left(h_v^{(k-1)}, m_{\mathcal{N}(v)}^{(k)}\right)$$

(Battaglia et al., 2016; Defferrard et al., 2016; Duvenaud et al., 2015; Hamilton et al., 2017a; Kearnes et al., 2016; Kipf & Welling, 2017; Li et al., 2016; Velickovic et al., 2018; Verma & Zhang, 2018; Ying et al., 2018b; Zhang et al., 2018)
Examples of aggregations

- Sum, average:

\[ m_{\mathcal{N}(v)} = \frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} h_u \]

Input node attribute of neighbors
Examples of aggregations

- Sum, average:
  \[ m_{\mathcal{N}(v)} = \frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} h_u \]

- General form (with small neural networks = MLPs):
  \[ m_{\mathcal{N}(v)} = \text{MLP}_\theta \left( \sum_{u \in \mathcal{N}(v)} \text{MLP}_\phi(h_u) \right) \]

- Update (combine):
  \[ h_v^{(k)} = \sigma \left( W_{\text{self}} h_u^{(k-1)} + W_{\text{neigh}} m_{\mathcal{N}(v)}^{(k)} + b \right) \]
Note: final “readout”

- The final readout is just another aggregation operation, parameterized separately

\[ h = \frac{1}{n} \sum_u h_u \]

1. **Node embeddings:**
   - encode the local neighborhood of each node separately via aggregations

2. **Readout:**
   - encode the entire graph by aggregating node embeddings
Node embeddings as neural network

(illustrations: J. Leskovec)
Node embeddings as neural network

(Illustrations: J. Leskovec)
Node embeddings as neural network

grey boxes: aggregation functions that we learn

(Illustrations: J. Leskovec)
Node embeddings as neural network

grey boxes: aggregation functions that we learn

(Illustrations: J. Leskovec)
Node embeddings as neural network

TARGET NODE

grey boxes: aggregation functions that we learn

INPUT GRAPH

(Illustrations: J. Leskovec)
Node embeddings as neural network

grey boxes: aggregation functions that we learn

(illustrations: J. Leskovec)
Node embeddings as neural network

Grey boxes: aggregation functions that we learn
Training a GNN

• What is a data point?
  
  \( \text{Node + its neighborhood} \)  
  \( \text{(computation tree)} \)

• What to specify?
  
  • Aggregate, combine and readout functions
  
  • Loss function on prediction

• Train with SGD
Summary: GNN model

- For many graph prediction problems, information about local neighborhoods adds important information.
- Encode local neighborhoods via a general form of “averaging”: aggregation.
- We learn the aggregation function via Stochastic Gradient Descent.

Comparison to CNN:
- Image = a graph where each node has the same number of neighbors.
- Both encode local neighborhoods.
- CNN does weighted average, GNN treats each neighbor the same.
Implementing GNNs

• **PyTorch Geometric:**
  [https://github.com/rusty1s/pytorch_geometric](https://github.com/rusty1s/pytorch_geometric)
  Introductory code example:

• **Deep Graph library:**
  [https://www.dgl.ai](https://www.dgl.ai)
  Introductory code example:
  [https://docs.dgl.ai/tutorials/basics/1_first.html](https://docs.dgl.ai/tutorials/basics/1_first.html)

Graph Convolutions: Initialization

<table>
<thead>
<tr>
<th>Atom</th>
<th>Degree</th>
<th>In Ring</th>
<th>Aromatic</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>1</td>
<td>FALSE</td>
<td>FALSE</td>
</tr>
</tbody>
</table>
# Real Initialization

<table>
<thead>
<tr>
<th>feature</th>
<th>description</th>
<th>size</th>
</tr>
</thead>
<tbody>
<tr>
<td>atom type</td>
<td>type of atom (ex. C, N, O), by atomic number</td>
<td>100</td>
</tr>
<tr>
<td># bonds</td>
<td>number of bonds the atom is involved in</td>
<td>6</td>
</tr>
<tr>
<td>formal charge</td>
<td>integer electronic charge assigned to atom</td>
<td>5</td>
</tr>
<tr>
<td>chirality</td>
<td>unspecified, tetrahedral CW/CCW, or other</td>
<td>4</td>
</tr>
<tr>
<td># Hs</td>
<td>number of bonded hydrogen atoms</td>
<td>5</td>
</tr>
<tr>
<td>hybridization</td>
<td>sp, sp2, sp3, sp3d, or sp3d2</td>
<td>5</td>
</tr>
<tr>
<td>aromaticity</td>
<td>whether this atom is part of an aromatic system</td>
<td>1</td>
</tr>
<tr>
<td>atomic mass</td>
<td>mass of the atom, divided by 100</td>
<td>1</td>
</tr>
</tbody>
</table>
Graph Convolutions: Local Aggregation
Graph Convolutions: Local Aggregation
Graph Convolutions: Message Propagation

2-hop neighborhood subgraph
Graph Convolutions: Molecular Representation

Toxic?
Performance Boosters

• Adding Expert Design Fingerprints (RDKIT)
  • Pay attention to scaling
  • Useful for small and or noisy datasets
• Hyperparameter Tuning
  • Bayesian Optimization
• Ensembling (for various random initializations)
Experimental Errors

Figure 9. Comparison of Amgen’s internal model and our D-MPNN (evaluated using a single run on a chronological split) to experimental error (higher = better). Note that the experimental error is not evaluated on the exact same time split as the two models since it can only be measured on molecules which were tested more than once, but even so the difference in performance is striking.
Splitting Strategies Vary

- Standard in Chemistry: scaffold split
- Split is driven by graph similarity
Scaffold Similarity

Tanimoto

Formula:

\[ \text{Sim}_{\text{Tanimoto}}(A, B) = \frac{\text{bothAB}}{|A| + |B| - \text{bothAB}} = \frac{\text{bothAB}}{\text{onlyA} + \text{onlyB} + \text{bothAB}} \]

Range:

[0.0, 1.0]

Example:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ \frac{\text{bothAB}}{\text{onlyA} + \text{onlyB} + \text{bothAB}} = \frac{3}{1 + 2 + 3} = \frac{3}{6} = 0.5 \]

Scaffold: molecular core to which functional groups are attached
Performance

(b) Classification Data Sets (higher = better).
Overlap in Molecular Scaffolds

The image shows a bar chart depicting the percent of molecules in test with scaffold in train for different categories: rPPB, Sol, RLM, and hPXR. The chart compares 'Random' and 'Time' scenarios with blue and red bars, respectively. The y-axis represents the percent of molecules, ranging from 0 to 100.
Performance as a function of the Split

![Graph showing performance metrics for rPPB, Sol, RLM, and hPXR]
Are We Making Progress?

Tox-21 scaffold split benchmark

<table>
<thead>
<tr>
<th>Method</th>
<th>% AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chemprop¹</td>
<td>77.5 ± 0.4</td>
</tr>
<tr>
<td>Chemprop Ensemble + FP</td>
<td>79.1 ± 0.1</td>
</tr>
<tr>
<td>Graph Isomorphism Network²</td>
<td>77.6 ± 0.6</td>
</tr>
<tr>
<td>Principal Neighborhood Aggregation³</td>
<td>80.7 ± 0.4</td>
</tr>
</tbody>
</table>

[2] Xu et al. How Powerful are Graph Neural Networks, 2019
Molecular Pretraining: Graveyard of Good Ideas

OLD SLIDE!
Models are Data Hungry

CIFAR-10 dataset
# classes: 10
# img per class: 5000

Test set accuracy

<table>
<thead>
<tr>
<th>Training Data</th>
<th>Test Set Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>50,000</td>
<td>95.16</td>
</tr>
<tr>
<td>100</td>
<td>18.22</td>
</tr>
</tbody>
</table>
Initialization via Pre-Training

Transfer knowledge acquired from other dataset

ImageNet dataset: 1.2M labeled images

Training
Weights in red are initialized from the pre-trained model and are fixed during fine tuning stage.

Weights in black are initialized from scratch and are updated by SGD during fine tuning stage.

Breast cancer detection dataset
Pre-training in NLP

- **Challenge:** less labeled data than CV
- **Solution:** masked language modeling with *lots* of unlabeled data (160 GB)

Devlin et al., BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019
Pre-training in NLP

GLUE benchmark

11 tasks for natural language understanding

100
87.5
75
62.5
50

Score

No Pretraining: 64.2
Human: 87.1
Pretraining: 88.5

Linguistic acceptability
Sentiment analysis
Paraphrase detection
Natural language inference
Question answering
Coreference resolution

Liu et al., RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019
Pre-training for Molecules: Chemistry Agnostic Approach

• **Idea:** words in sentence ⇔ atoms in molecule
  structural similarity of nodes’ network neighborhoods

![Diagram showing GNN (Graph Neural Network) with masked nodes and substructure context]

a) predicting masked atom in molecule   b) predicting context from substructure

• **Data:** 2M molecules from ZINC

No consistent improvement across datasets!
Pre-training for Molecules: Utilizing Chemical Reactions

• **Idea:** preserve the equivalence of molecular representation wrt to chemical reactions

  sum of reactants embeddings $\iff$ product embedding

• **Data:** USPT0-479K dataset
Pre-training for Molecules: Utilizing Chemical Reactions

<table>
<thead>
<tr>
<th>Datasets</th>
<th>BBBP</th>
<th>HIV</th>
<th>BACE</th>
<th>Tox21</th>
<th>ClinTox</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMILES-Transformers</td>
<td>0.704</td>
<td>0.729</td>
<td>0.701</td>
<td>0.802</td>
<td>0.954</td>
</tr>
<tr>
<td>ECFP4</td>
<td>0.729</td>
<td>0.792</td>
<td>0.867</td>
<td>0.822</td>
<td>0.799</td>
</tr>
<tr>
<td>GraphConv</td>
<td>0.690</td>
<td>0.763</td>
<td>0.783</td>
<td>0.829</td>
<td>0.807</td>
</tr>
<tr>
<td>Weave</td>
<td>0.671</td>
<td>0.703</td>
<td>0.806</td>
<td>0.820</td>
<td>0.832</td>
</tr>
<tr>
<td>ChemBERTa</td>
<td>0.643</td>
<td>0.622</td>
<td>-</td>
<td>0.728</td>
<td>0.733</td>
</tr>
<tr>
<td>D-MPNN</td>
<td>0.708</td>
<td>0.752</td>
<td>-</td>
<td>0.688</td>
<td>0.906</td>
</tr>
<tr>
<td>CDDD</td>
<td>0.761 ± 0.00</td>
<td>0.753 ± 0.00</td>
<td>0.833 ± 0.00</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MolBERT</td>
<td>0.762 ± 0.00</td>
<td>0.783 ± 0.00</td>
<td>0.866 ± 0.00</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Mol2vec</td>
<td>0.872 ± 0.021</td>
<td>0.769 ± 0.021</td>
<td>0.862 ± 0.027</td>
<td>0.803 ± 0.041</td>
<td>0.841 ± 0.062</td>
</tr>
<tr>
<td>MolR-GCN</td>
<td>0.890 ± 0.032</td>
<td><strong>0.802 ± 0.024</strong></td>
<td><strong>0.882 ± 0.019</strong></td>
<td>0.818 ± 0.023</td>
<td>0.916 ± 0.039</td>
</tr>
<tr>
<td>MolR-GAT</td>
<td>0.887 ± 0.026</td>
<td>0.794 ± 0.022</td>
<td>0.863 ± 0.026</td>
<td><strong>0.839 ± 0.039</strong></td>
<td>0.908 ± 0.039</td>
</tr>
<tr>
<td>MolR-SAGE</td>
<td>0.879 ± 0.032</td>
<td>0.793 ± 0.026</td>
<td>0.859 ± 0.029</td>
<td>0.811 ± 0.039</td>
<td>0.890 ± 0.058</td>
</tr>
<tr>
<td>MolR-TAG</td>
<td><strong>0.895 ± 0.031</strong></td>
<td>0.801 ± 0.023</td>
<td>0.875 ± 0.023</td>
<td>0.820 ± 0.028</td>
<td>0.913 ± 0.043</td>
</tr>
</tbody>
</table>

Chemical-Reaction-Aware Molecule Representation Learning.
Hongwei Wang, Weijiang Li, Xiaomeng Jin, Kyunghyun Cho, Jiawei Han, Martin D Burke, Heng Ji, ICLR, 2022.
Effective against drug resistant strains

E. coli
C. difficile
A. baumannii

Effective against drug resistant strains
Narrow Spectrum Antibiotics: Abacin
Discovery on the Budget

Train: 2.5K molecules screened against E.coli

Antibacterial activity

Jon Stokes
New antibiotic discovery

E. coli growth inhibition training set (~2K)

GNN model (ensemble)

chemical space screening

validation of hits against multiple bacterial species

ZINC (10^8)

Repurposing hub (10^4)

…

[Stokes et al. 2020]
Driving Forces behind the Finding

Predictor accuracy — 51% on top 100 molecules

- Smart Manual Selection of Training/Testing Split
- Smart Manual Filtering of Candidates

While neural network models narrowed the performance gap between analytical and experimental approaches, a difference still exists. Here, we demonstrate how the combination of \textit{in silico} predictions and empirical investigations can lead to the discovery of new antibiotics (Figure 1). Our approach consists of three stages. First, we trained a deep neural network model to predict growth inhibition of \textit{Escherichia coli} using a collection of 2,335 molecules. Second, we applied the resulting
New antiviral combinations

- A perfect single drug may not always be available or sufficiently effective, e.g., against SARS-COV-2 infection
- Drug combination therapies that tap into multiple parts of viral infection can be substantially more effective than individual drugs (e.g., HIV)
- We aim to find drugs that synergistically inhibit SARS-COV-2 infection
  - e.g., viral entry into the cell
  - e.g., viral replication processes
  - e.g., interaction with host proteins

Specific challenge: combination data is extremely limited

Figure source: Cevik et al., BMJ 2020
**Goal**: build both accurate and interpretable models to predict synergistic effects of drug combinations for SARS-CoV-2

**Data**: individual drug effectiveness (~10K, few positives), drug-target interaction data (~20K), very limited combination screens (~160 NCATS)
• **Goal:** build both accurate and interpretable models to predict synergistic effects of drug combinations for SARS-CoV-2

• **Data:** individual drug effectiveness (~10K, few positives), drug-target interaction data (~ 20K), very limited combination screens (~160 NCATS)
**Goal:** build both accurate and interpretable models to predict synergistic effects of drug combinations for SARS-CoV-2

**Data:** individual drug effectiveness (≈10K, few positives), drug-target interaction data (≈ 20K), very limited combination screens (≈160 NCATS)
**Goal:** build both accurate and interpretable models to predict synergistic effects of drug combinations for SARS-CoV-2

**Data:** individual drug effectiveness (~10K, few positives), drug-target interaction data (~ 20K), very limited combination screens (~160 NCATS)
New combination discoveries

- Experimentally tested 30 drug high ranking candidate combinations in NCATS SARS-Cov-2 CPE assay

**Remdesivir + Reserpine**

Both drugs are weak individually at 10μM (>40% virus alive)

The combination is potent at 3μM (only 3.2% virus alive)

**Remdesivir + IQ-1S**

Both drugs are weak individually at 10μM (>60% virus alive)

The combination is potent at 3μM (0% virus alive)

[Jin et al. 2020d]