ChemicalX

Deep Learning For Drug Pair Scoring

EN104 – Advanced Deep Learning 2024

Edwin Tembo

Professor Zoran B. Djordjevic

Harvard Extension School

The Team

AstraZeneca

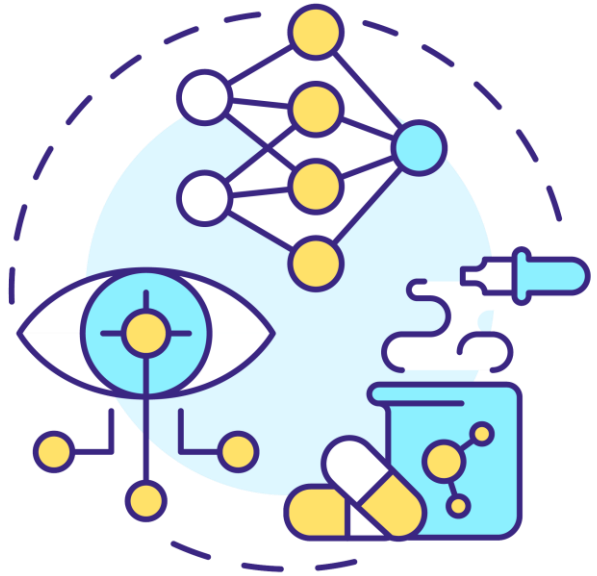
Vanderbilt University

The Problem : Drug Pair Scoring

+ Status Quo :

Weeks/Months/Years

+ Goal :

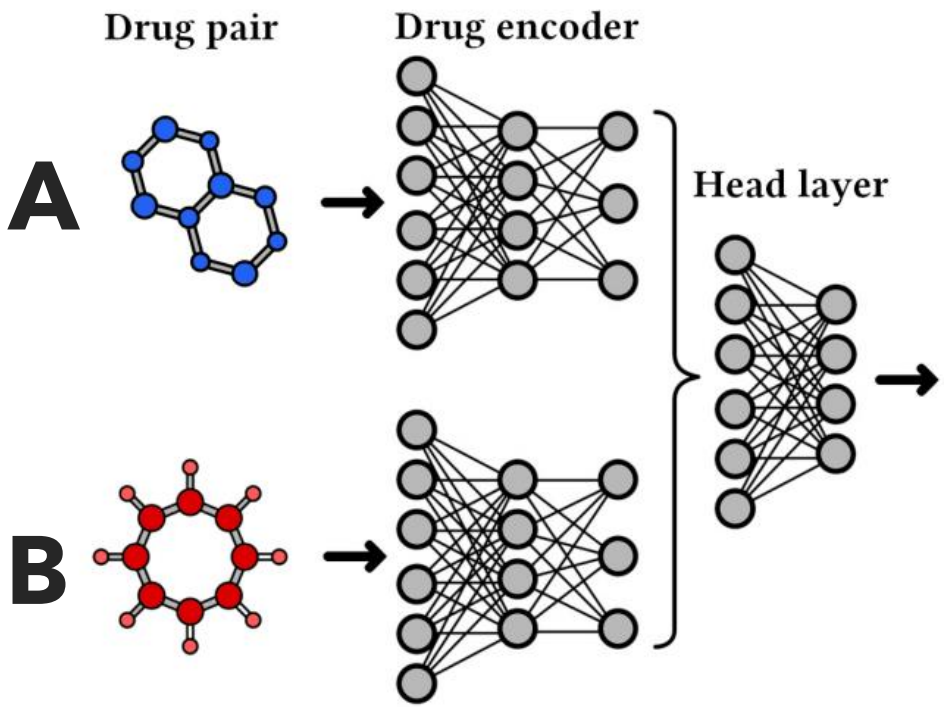


Minutes/Hours/Days

Drug Pair Scoring

- Drug-Drug Interaction
- Polypharmacy
- Drug Synergy

Model Architecture



Probability Of :

- **Polypharmacy side effect Z**
- **Interaction Z**
- **Synergy at treating disease Z**

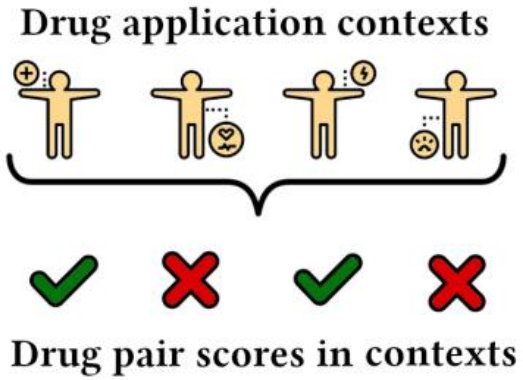
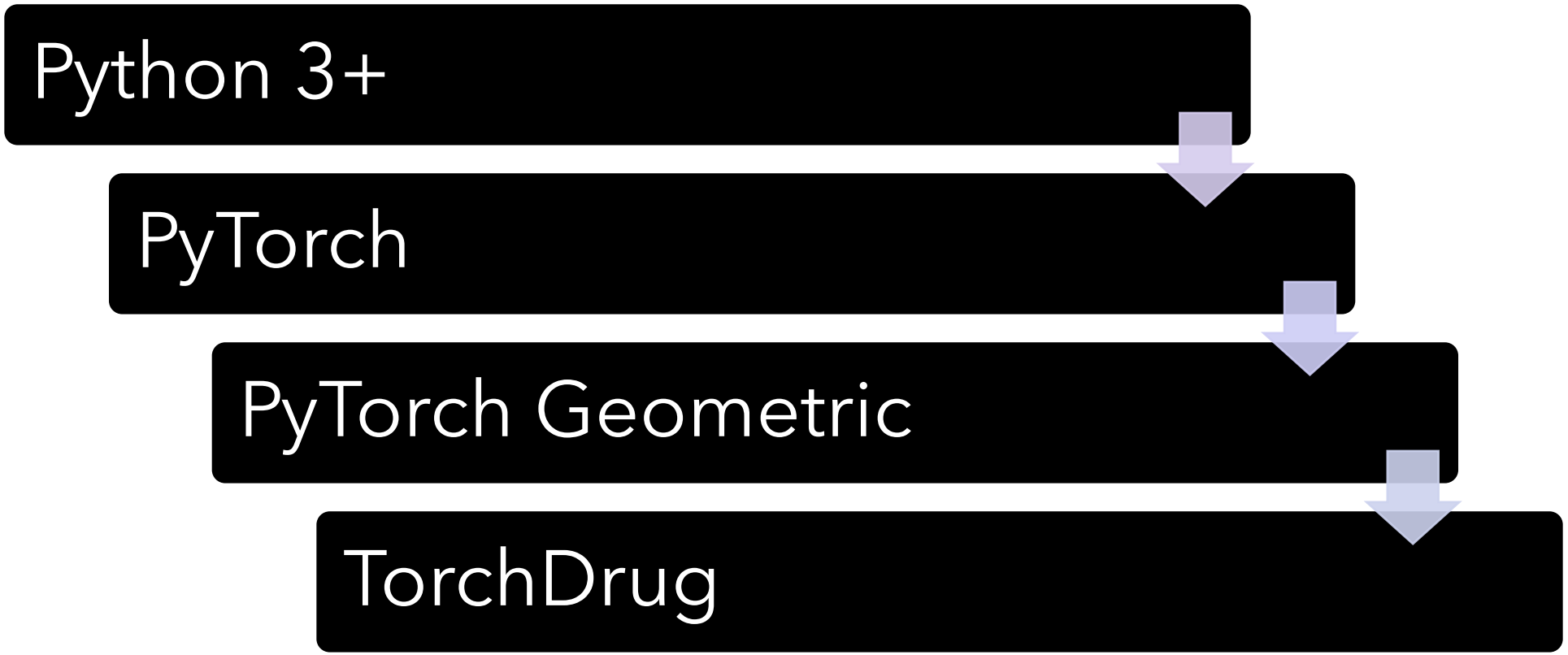


Image Source: ChemicalX Paper

Main Requirements:



Datasets

Dataset	Task
TWOSIDES [44]	Polypharmacy
Drugbank DDI [41]	Interaction
DrugComb [51, 52]	Synergy
DrugCombDB [28]	Synergy
OncolyPharm [21]	Synergy

Image Source: ChemicalX Paper

Preprocessed Data Structure

FeatureSet:

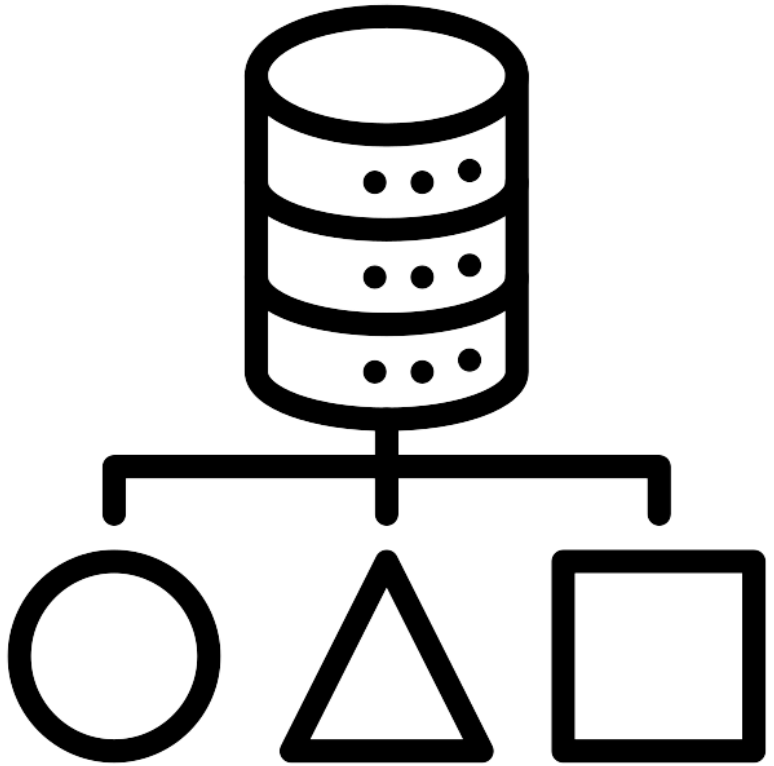
- 256 dimensional hashed Morgan fingerprints for each drug from SMILES.
- TorchDrug Molecular Graph

ContextSet:

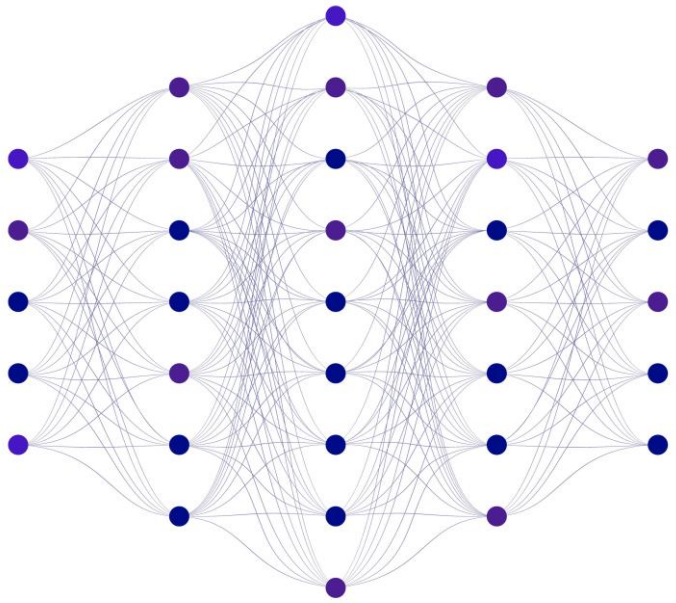
- An encoded context, either a biological aspect such as a cancer cell or another chemical compound.

Labeled Tripple :

- Identifiers for each drug in separate columns, with a classification label in 3rd column.



Models



Model	Year	Domain	Encoder
DeepDDI [41]	2018	Interaction	Feedforward
DeepSynergy [37]	2018	Synergy	Feedforward
MHCADDI [9]	2019	Polypharmacy	GAT
MR-GNN [50]	2019	Interaction	GCN
CASTER [22]	2019	Interaction	Feedforward
SSI-DDI [34]	2020	Interaction	GAT
EPGCN-DS [43]	2020	Interaction	GCN
DeepDrug [4]	2020	Interaction	GCN
GCN-BMP [7]	2020	Interaction	GCN
DeepDDS [47]	2021	Synergy	GCN or GAT
MatchMaker [3]	2021	Synergy	Feedforward

Image Source: ChemicalX Paper

Utilities

Abstractions provided include:

Dataloader

BatchGenerator

Model Pipeline for Training/Inference

Sample Code: Batch Generator

```
from chemicalx.data import DrugCombDB, BatchGenerator

loader = DrugCombDB()

context_set = loader.get_context_features()
drug_set = loader.get_drug_features()
triples = loader.get_labeled_triples()

train, test = triples.train_test_split(train_size=0.5)

generator = BatchGenerator(batch_size=1024,
                           context_features=True,
                           drug_features=True,
                           drug_molecules=False,
                           context_feature_set=context_set,
                           drug_feature_set=drug_set,
                           labeled_triples=train)
```

Sample Code: Regular Training

```
1
2  import torch
3  from chemicalx.models import DeepSynergy
4
5  model = DeepSynergy(context_channels=112,
6                      drug_channels=256)
7
8  optimizer = torch.optim.Adam(model.parameters())
9  model.train()
10 loss = torch.nn.BCELoss()
11
12 for batch in generator:
13     optimizer.zero_grad()
14     prediction = model(batch.context_features,
15                        batch.drug_features_left,
16                        batch.drug_features_right)
17     loss_value = loss(prediction, batch.labels)
18     loss_value.backward()
19     optimizer.step()
```

Image Source: ChemicalX Paper

Sample Code: Pipeline Training

```python
from chemicalx import pipeline
from chemicalx.models import DeepSynergy
from chemicalx.data import DrugCombDB

model = DeepSynergy(context_channels=112, drug_channels=256)
dataset = DrugCombDB()

results = pipeline(
    dataset=dataset,
    model=model,
    # Data arguments
    batch_size=5120,
    context_features=True,
    drug_features=True,
    drug_molecules=False,
    # Training arguments
    epochs=100,
)

# Outputs information about the AUC-ROC, etc. to the console.
results.summarize()

# Save the model, losses, evaluation, and other metadata.
results.save("/content/test_results/")
```

Sample Code: Inference

```
import pandas as pd

model.eval()
generator.labeled_triples = test

predictions = []
for batch in generator:
    prediction = model(batch.context_features,
                       batch.drug_features_left,
                       batch.drug_features_right)
    prediction = prediction.detach().cpu().numpy()
    identifiers = batch.identifiers
    identifiers["prediction"] = prediction
    predictions.append(identifiers)
predictions = pd.concat(predictions)
```

Image Source: ChemicalX Paper