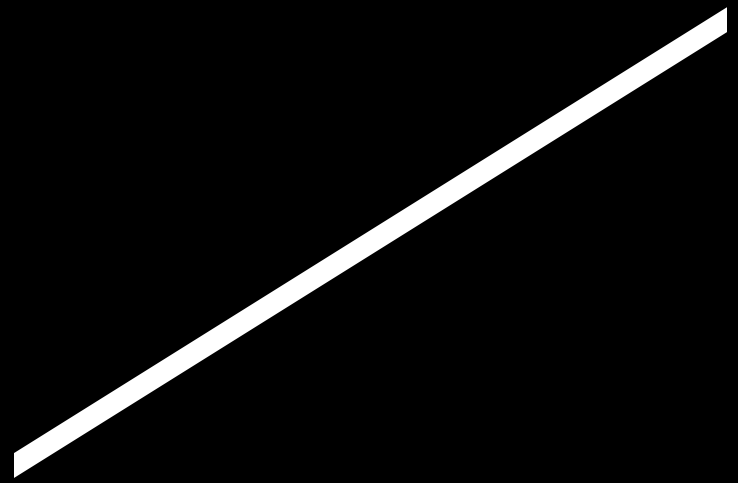
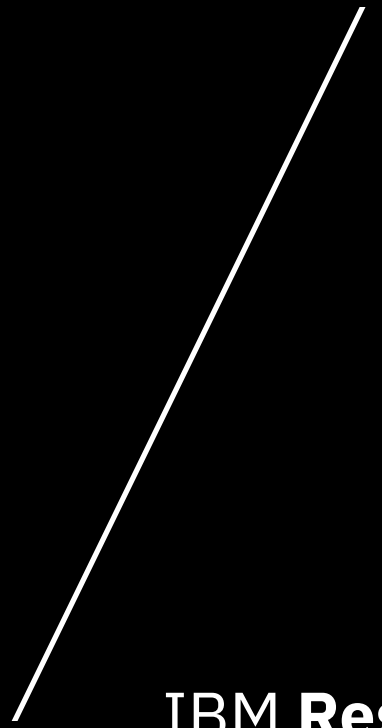
Accelerating Science with the Generative Toolkit for Scientific Discovery (GT4SD)

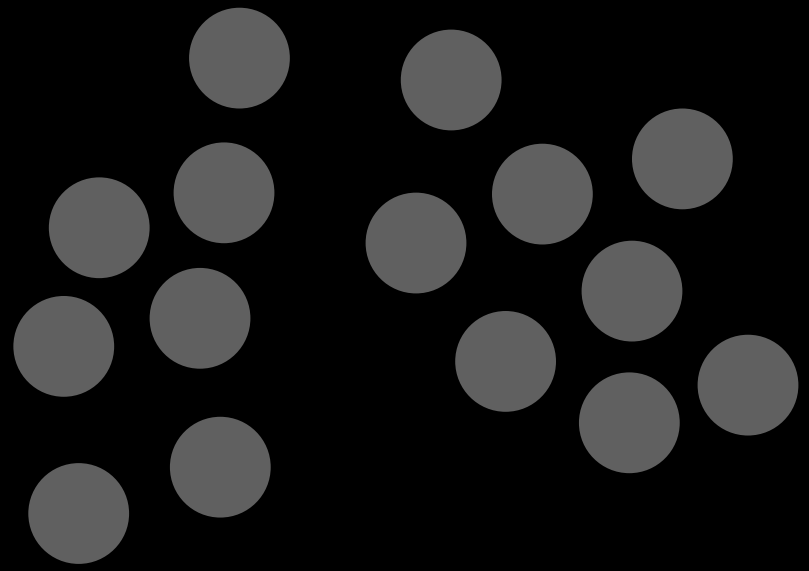




GT4SD/gt4sd-core

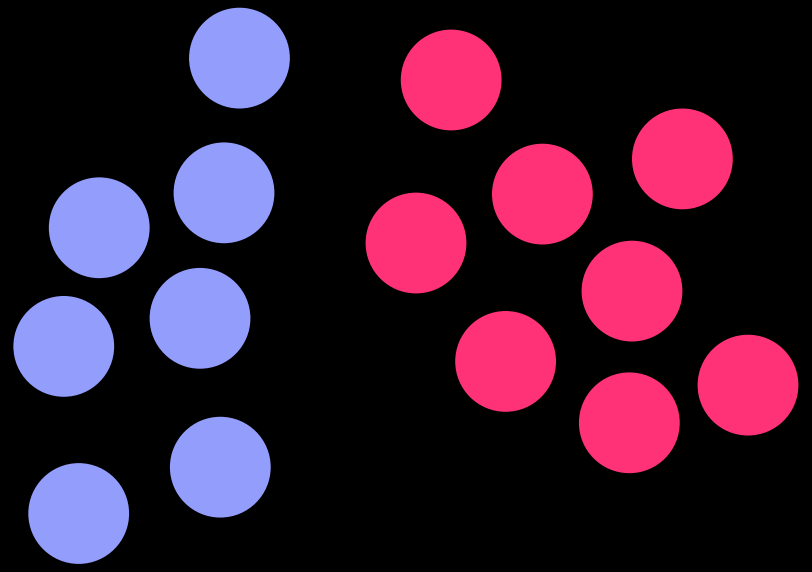
IBM **Research**

Introduction to generative models



Introduction to generative models

y = “active”

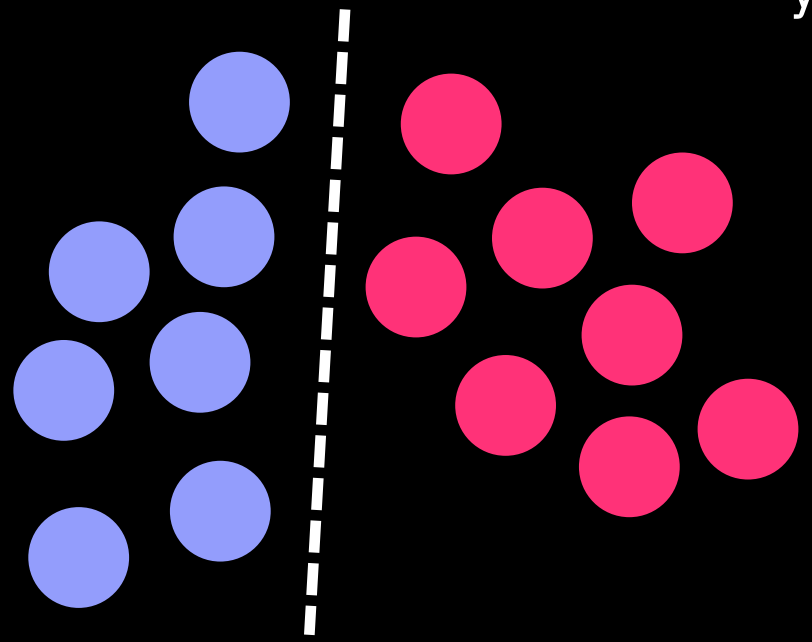


y = “inactive”

Considering data points (**X**) and associated labels (**Y**)

Introduction to generative models

y = “active”

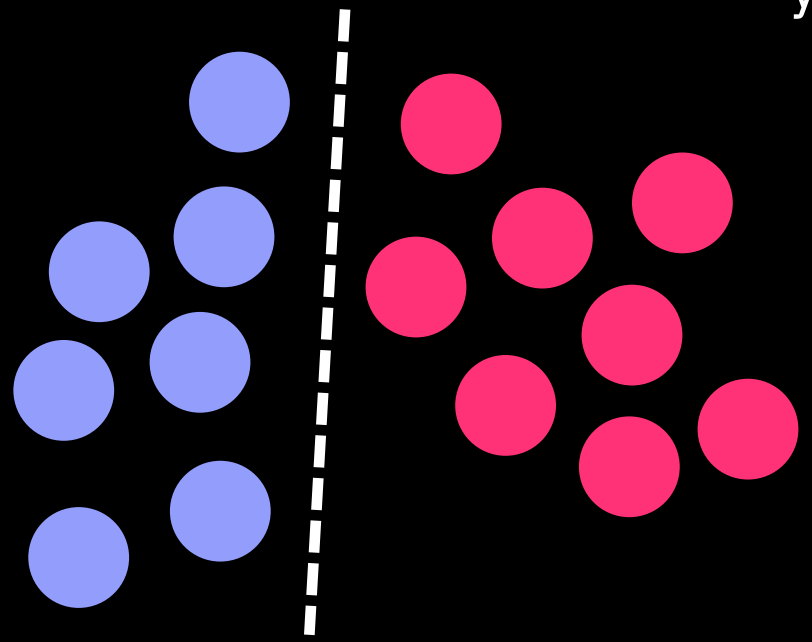


y = “inactive”

Discriminative models can learn boundaries

Introduction to generative models

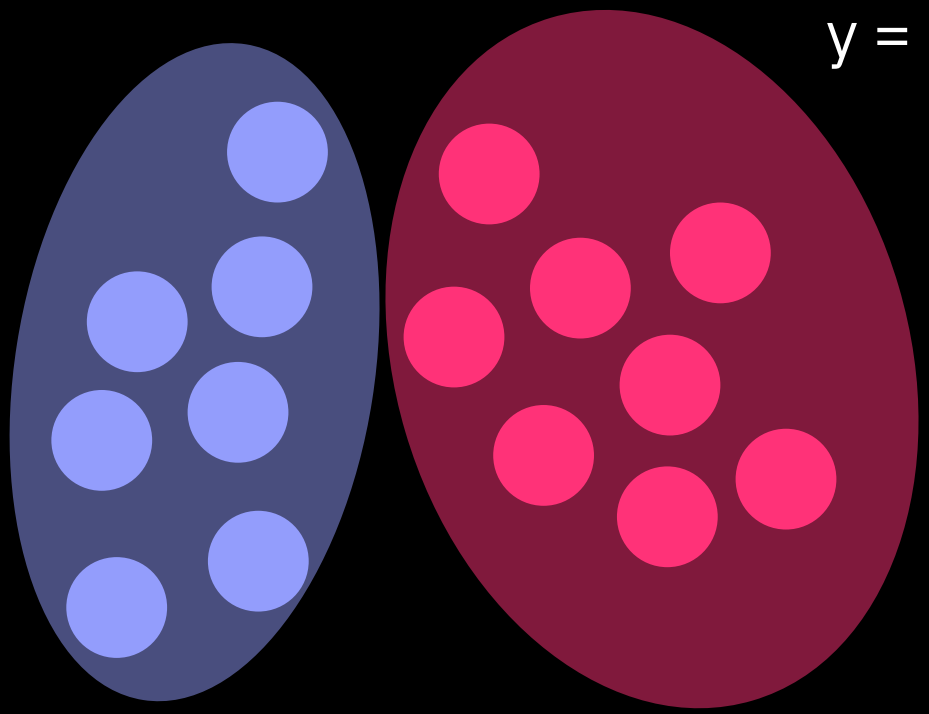
y = “active”



y = “inactive”

Discriminative models can learn **p(Y | X)**

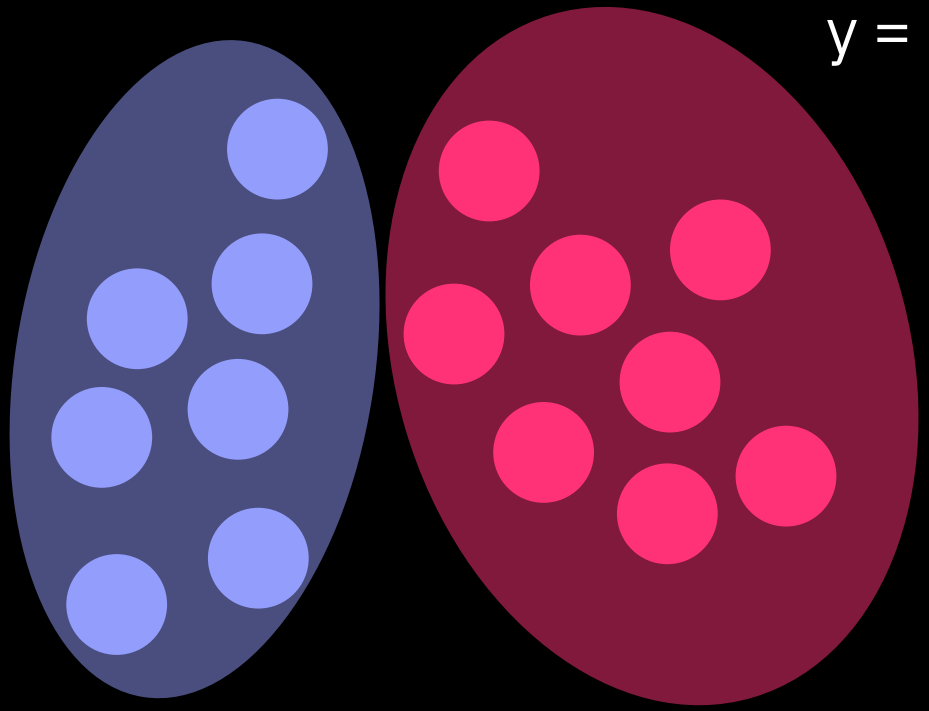
Introduction to generative models



Generative can learn joint data-labels distributions

Introduction to generative models

y = “active”

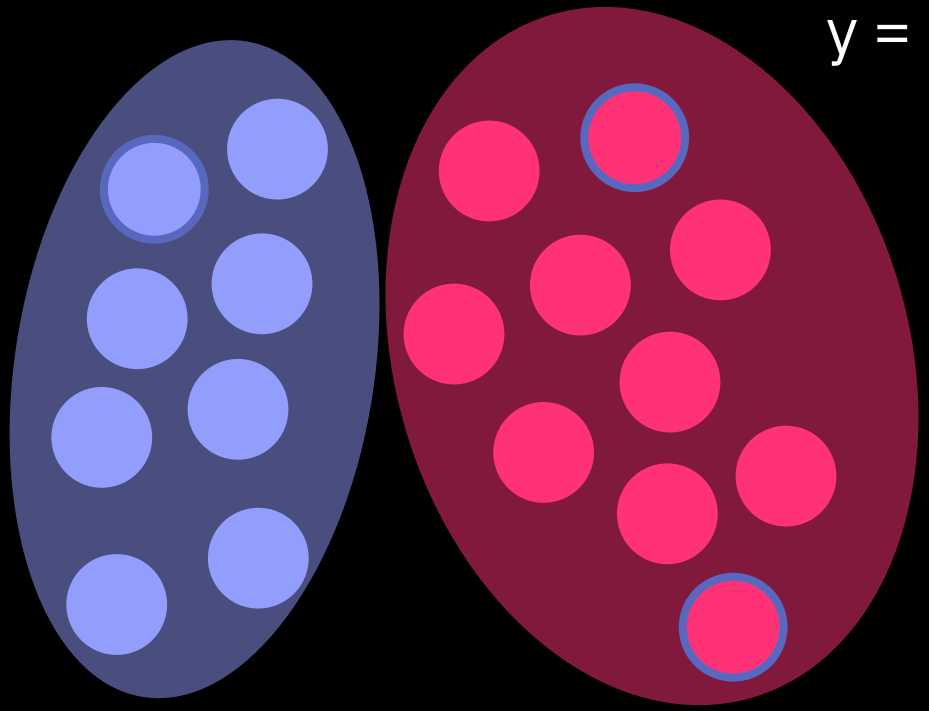


y = “inactive”

Generative models learn **p(X, Y)**

Introduction to generative models

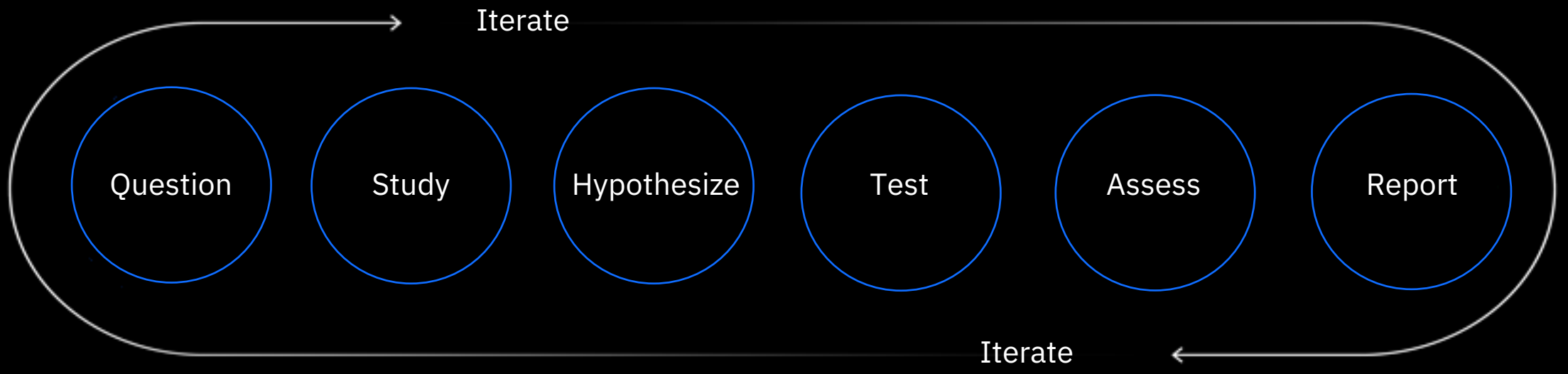
y = “active”



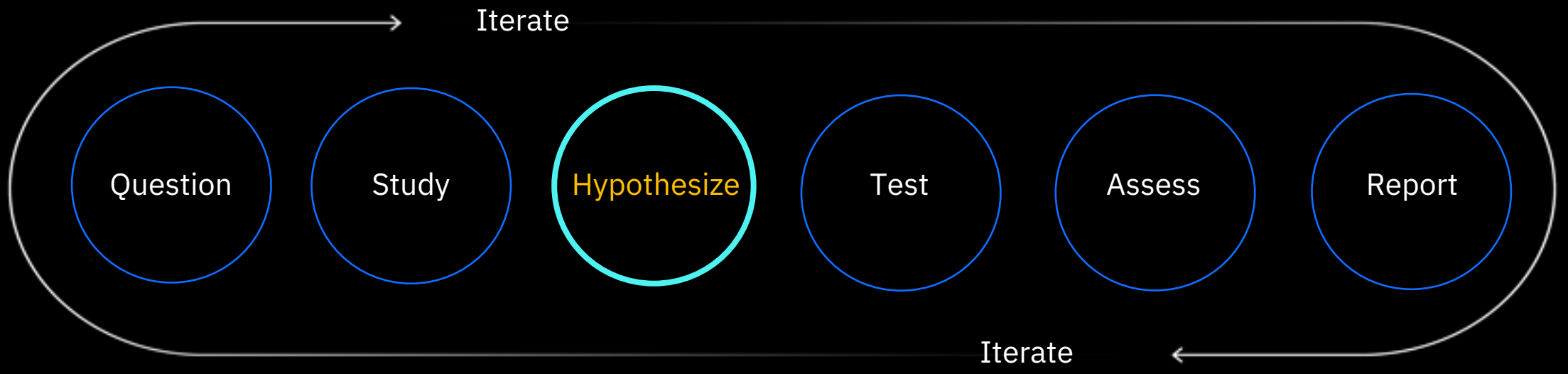
y = “inactive”

This allows to generate and create novel instances sampling from the underlying distributions

Generative models have the power to supercharge various aspects of the scientific method



We decided to start focusing on the step of accelerating hypothesis generation



Generative Toolkit for Scientific Discovery (GT4SD)

GT4SD makes generative AI algorithms and models easier to use in scientific discovery

GT4SD/gt4sd-core

Study → Hypothesize → Test

Applications include hypothesis generation for inverse design and discovery of materials (25+ models are available)

1. Train generative models

```
gt4sd-trainer  --training_pipeline_name paccmann-vae-trainer --epochs 25
```

2. Create inference pipelines

```
gt4sd-saving --training_pipeline_name paccmann-vae-trainer --model_path
```

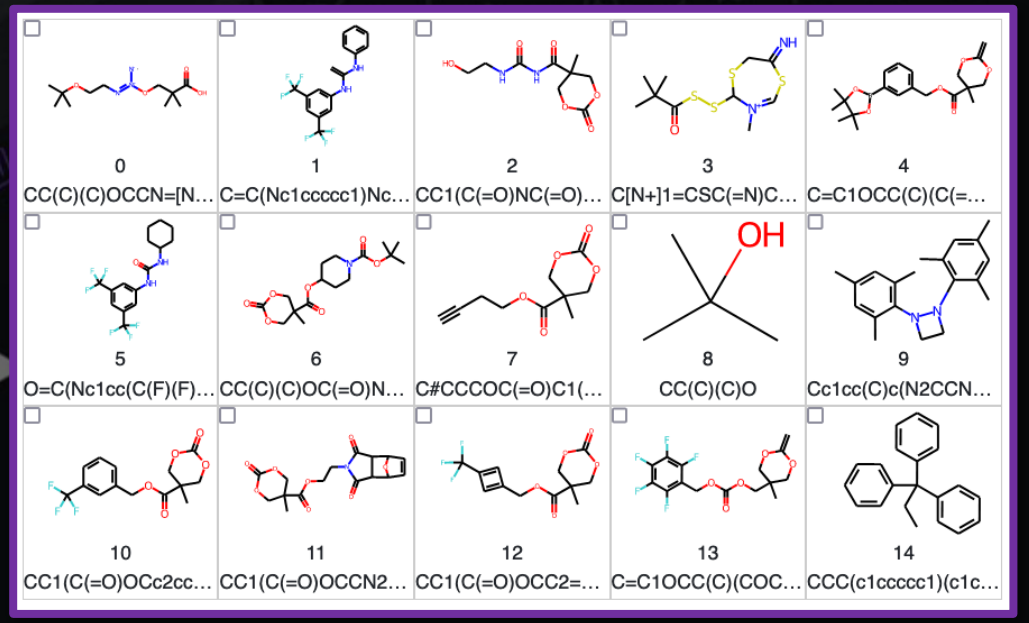
3. Run inference pipelines

```
gt4sd-inference --algorithm_name PaccMannGP --algorithm_application Pacc
```

4. Share your models with the community

```
gt4sd-upload --training_pipeline_name paccmann-vae-trainer --model_path
```

Molecules generated with GT4SD workflows



Generative Toolkit for Scientific Discovery (GT4SD)

GT4SD makes generative AI algorithms and models easier to use in scientific discovery

GT4SD/gt4sd-core

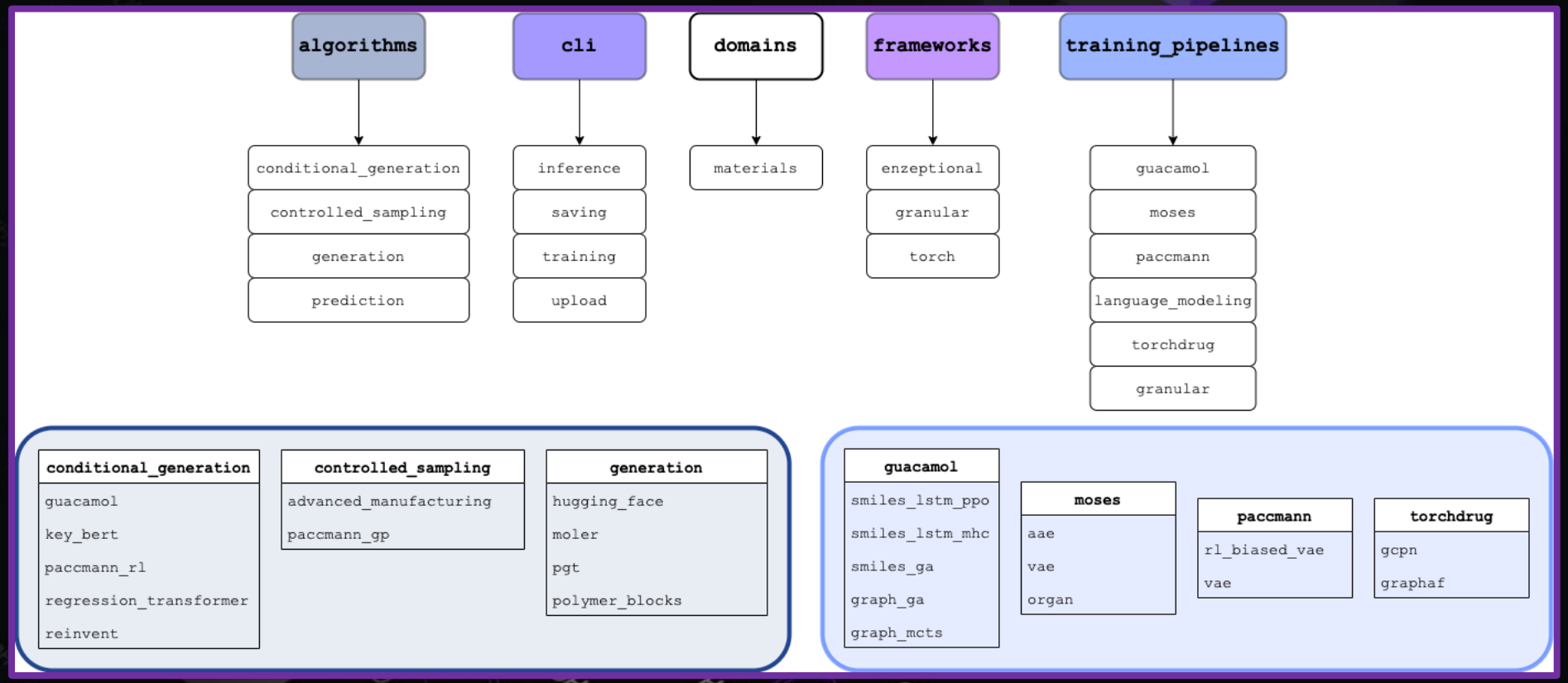

```python
from gt4sd.algorithms.registry import ApplicationsRegistry
# target definition (can be None)
target = ...
algorithm = ApplicationsRegistry.get_application_instance(
    target=target,
    algorithm_type="algorithm_type",
    domain="materials",
    algorithm_name="AlgorithmName",
    algorithm_application="AlgorithmApplication"
    # include additional configuration parameters as **kwargs
)
# get 50 samples from the algorithm
generated_samples = list(algorithm.sample(50))
```

```bash
# train a generative algorithm
gt4sd-trainer --training_pipeline_name ${TRAINING_PIPELINE_NAME} \
    --a_parameter 250 --another_parameter 4 --batch_size

# save it for usage
gt4sd-saving --training_pipeline_name ${TRAINING_PIPELINE_NAME} \
   --a_path /path/to/artifacts --target_version v1 \
   --algorithm_application AlgorithmApplication

# generate samples
gt4sd-inference --algorithm_name AlgorithmName \
    --algorithm_application AlgorithmApplication \
    --algorithm_version v1 --number_of_samples 25
```

Generative Toolkit for Scientific Discovery (GT4SD)

GT4SD makes generative AI algorithms and models easier to use in scientific discovery

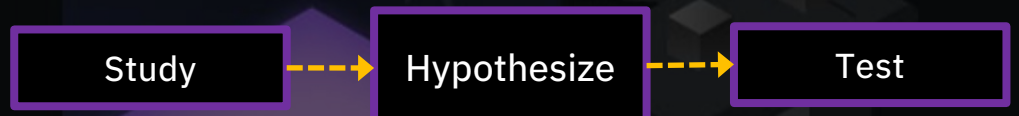
GT4SD/gt4sd-core

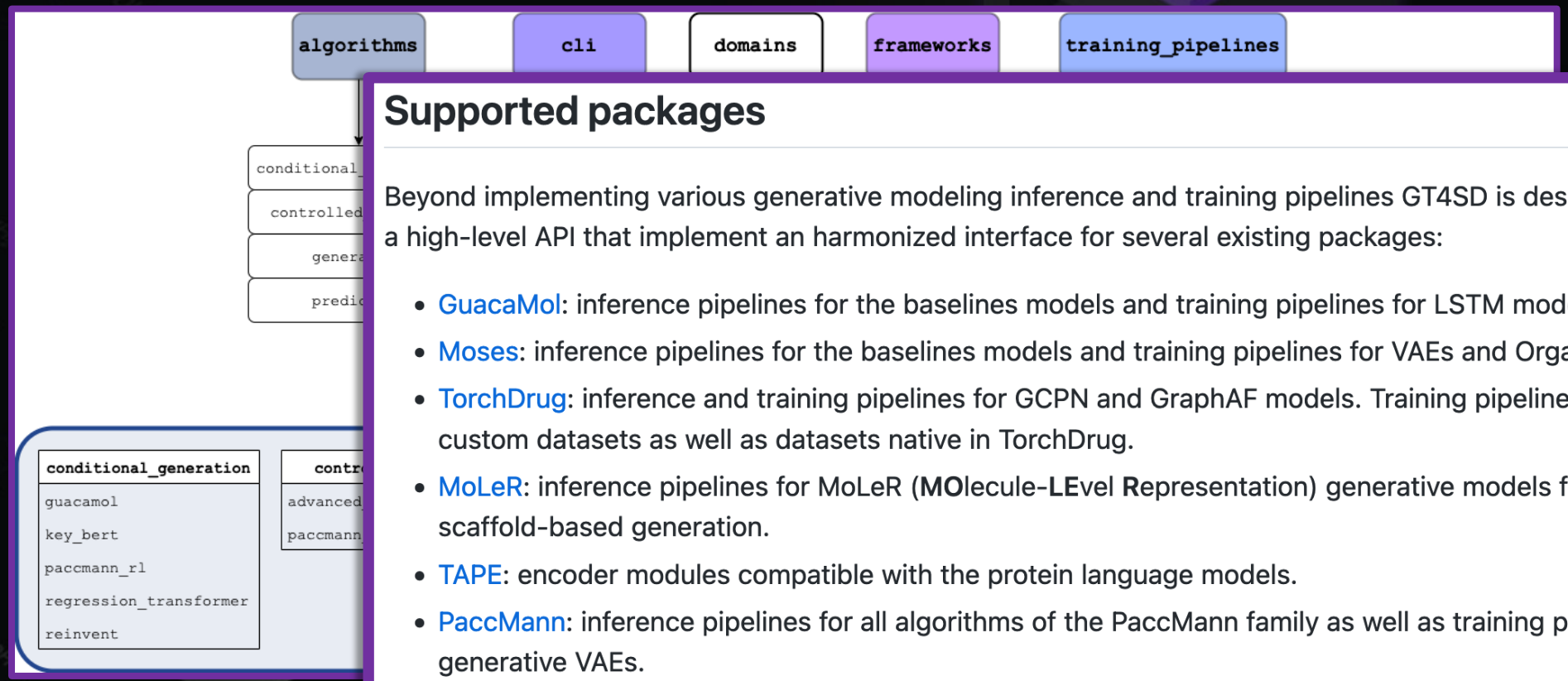


Generative Toolkit for Scientific Discovery (GT4SD)

GT4SD makes generative AI algorithms and models easier to use in scientific discovery

GT4SD/gt4sd-core





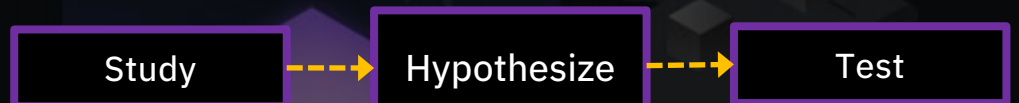
Supported packages

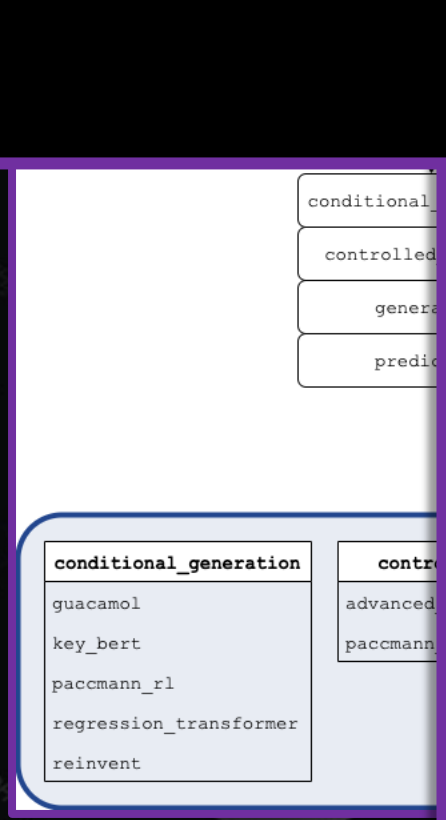
Beyond implementing various generative modeling inference and training pipelines GT4SD is designed to provide a high-level API that implement an harmonized interface for several existing packages:

- GuacaMol: inference pipelines for the baselines models and training pipelines for LSTM models.
- Moses: inference pipelines for the baselines models and training pipelines for VAEs and Organ.
- TorchDrug: inference and training pipelines for GCPN and GraphAF models. Training pipelines support custom datasets as well as datasets native in TorchDrug.
- MoLeR: inference pipelines for MoLeR (**MO**lecule-**LE**vel **R**epresentation) generative models for de-novo and scaffold-based generation.
- TAPE: encoder modules compatible with the protein language models.
- PaccMann: inference pipelines for all algorithms of the PaccMann family as well as training pipelines for the generative VAEs.
- transformers: training and inference pipelines for generative models from HuggingFace Models

Generative Toolkit for Scientific Discovery (GT4SD)

Textual representation of molecules (SMILES/SELFIES) combined w/ GAN/VAE/GA



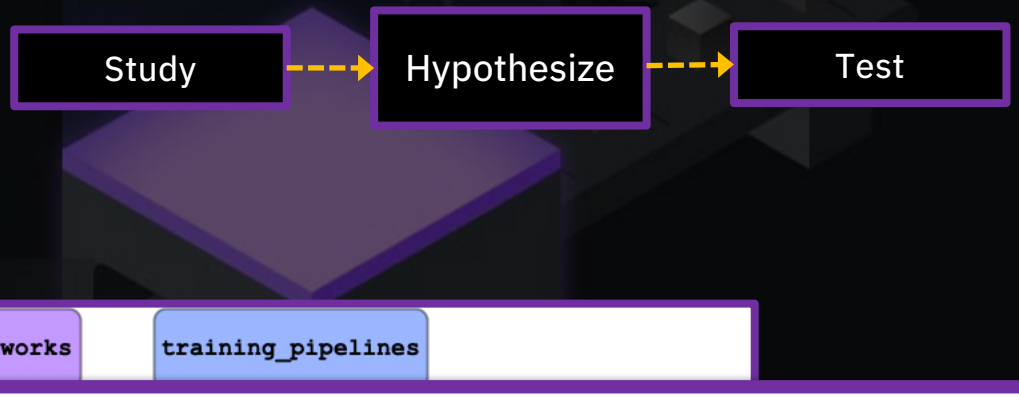


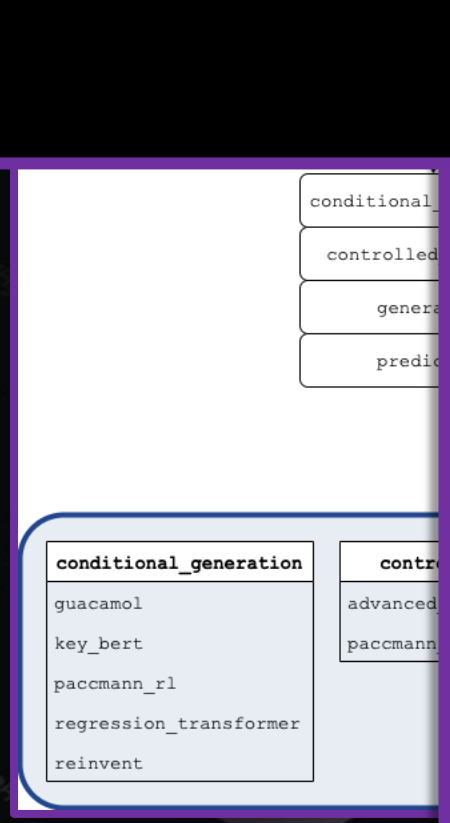
Beyond implementing various generative modeling inference and training pipelines GT4SD is designed to provide a high-level API that implement an harmonized interface for several existing packages:

- GuacaMol: inference pipelines for the baselines models and training pipelines for LSTM models.
- Moses: inference pipelines for the baselines models and training pipelines for VAEs and Organ.
- TorchDrug: inference and training pipelines for GCPN and GraphAF models. Training pipelines support custom datasets as well as datasets native in TorchDrug.
- MoLeR: inference pipelines for MoLeR (**MO**lecule-**LE**vel **R**epresentation) generative models for de-novo and scaffold-based generation.
- TAPE: encoder modules compatible with the protein language models.
- PaccMann: inference pipelines for all algorithms of the PaccMann family as well as training pipelines for the generative VAEs.
- transformers: training and inference pipelines for generative models from HuggingFace Models

Generative Toolkit for Scientific Discovery (GT4SD)

Graph representation of molecules combined w/ GCN/VAE/GA

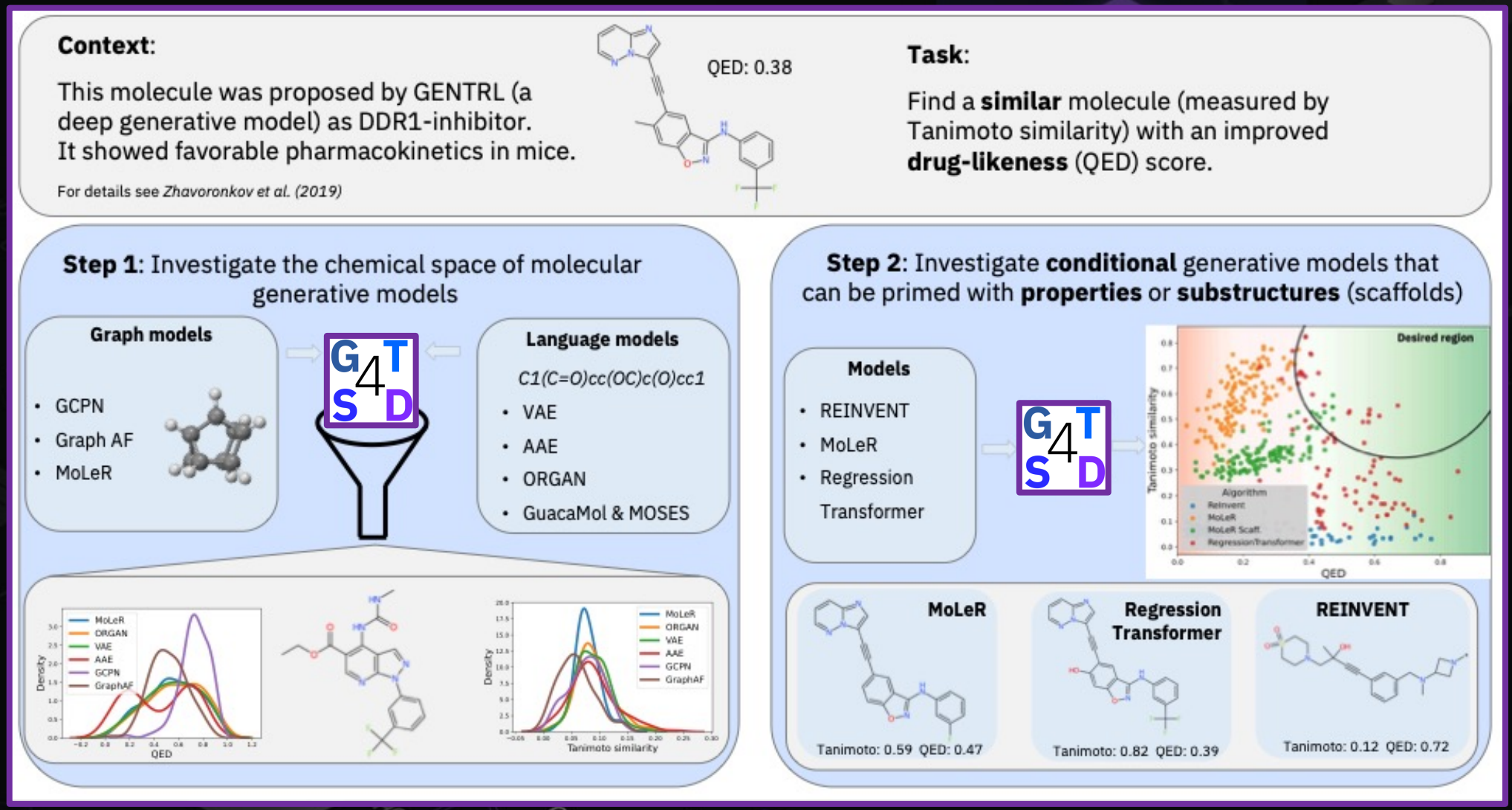




Beyond implementing various generative modeling inference and training pipelines GT4SD is designed to provide a high-level API that implement an harmonized interface for several existing packages:

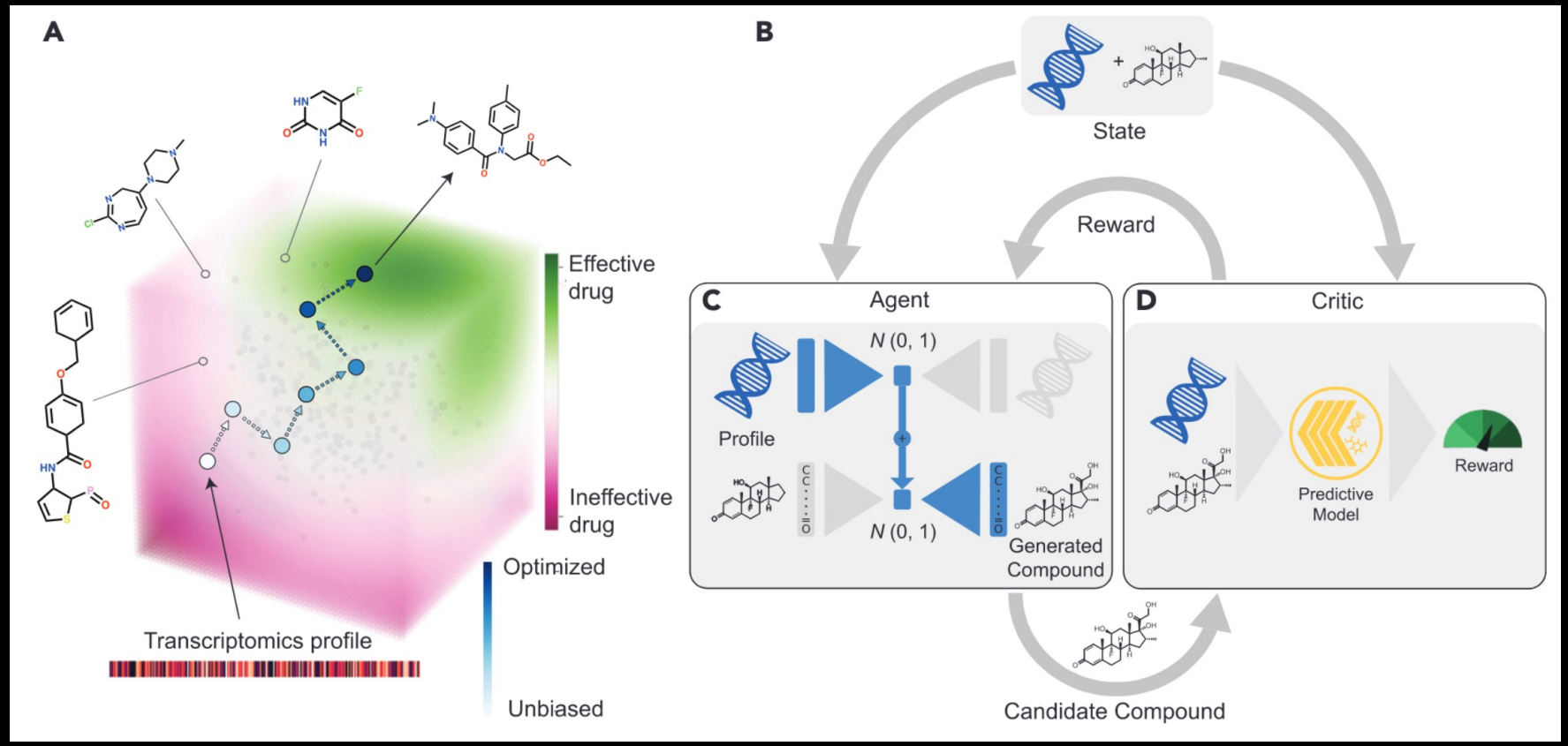
- GuacaMol: inference pipelines for the baselines models and training pipelines for LSTM models.
- Moses: inference pipelines for the baselines models and training pipelines for VAEs and Organ.
- TorchDrug: inference and training pipelines for GCPN and GraphAF models. Training pipelines support custom datasets as well as datasets native in TorchDrug.
- MoLeR: inference pipelines for MoLeR (**MO**lecule-**LE**vel **R**epresentation) generative models for de-novo and scaffold-based generation.
- TAPE: encoder modules compatible with the protein language models.
- PaccMann: inference pipelines for all algorithms of the PaccMann family as well as training pipelines for the generative VAEs.
- transformers: training and inference pipelines for generative models from HuggingFace Models

GT4SD enables complete discovery workflows



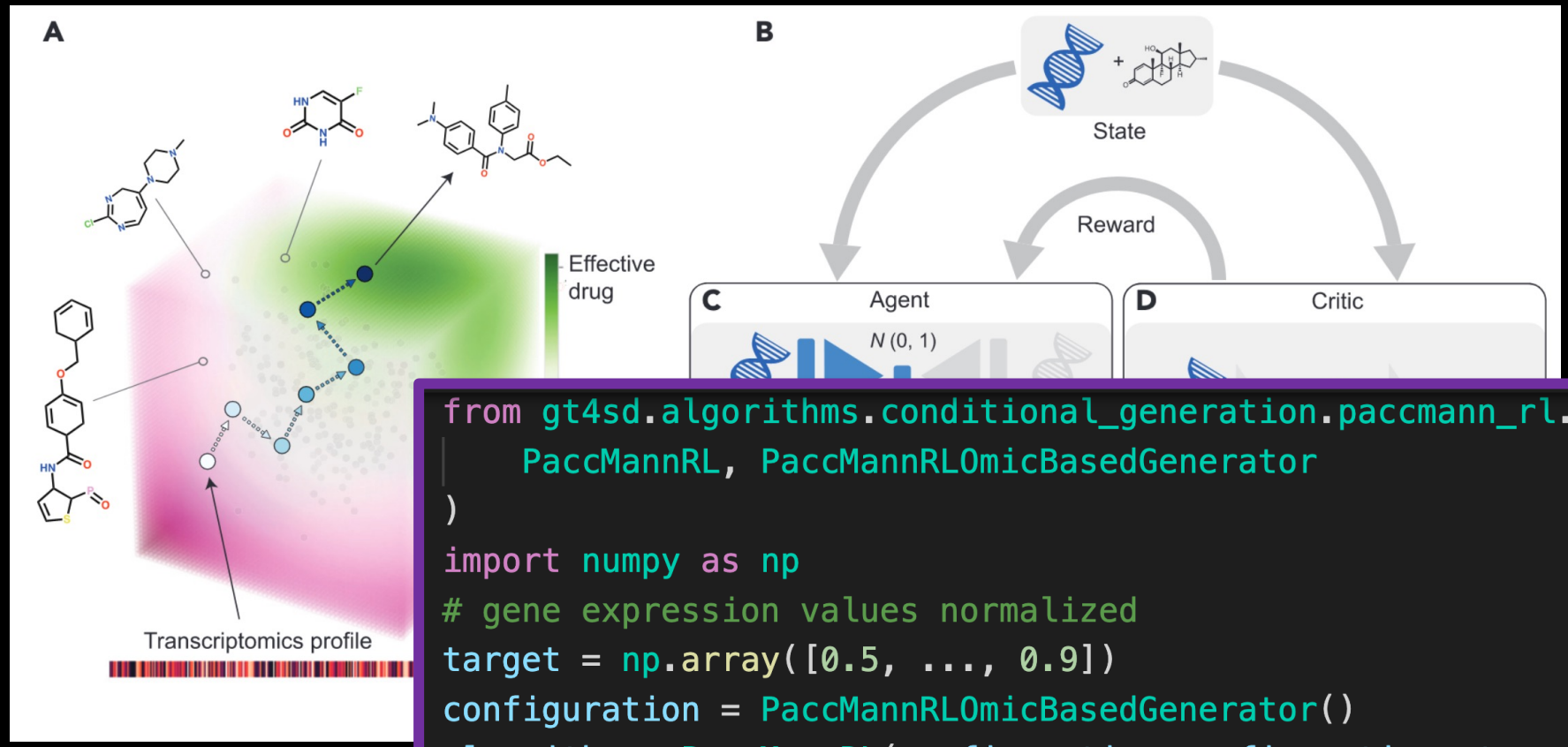
https://github.com/GT4SD/gt4sd-core/blob/main/notebooks/discovery-demo.ipynb

String-based (1D) Generative Molecule Creation - Gene Expression-driven Hit-like Molecule Design



Born *et al.* (2021). PaccMannRL: De novo generation of hit-like anticancer molecules from transcriptomic data via reinforcement learning. *iScience*, 24, 4.

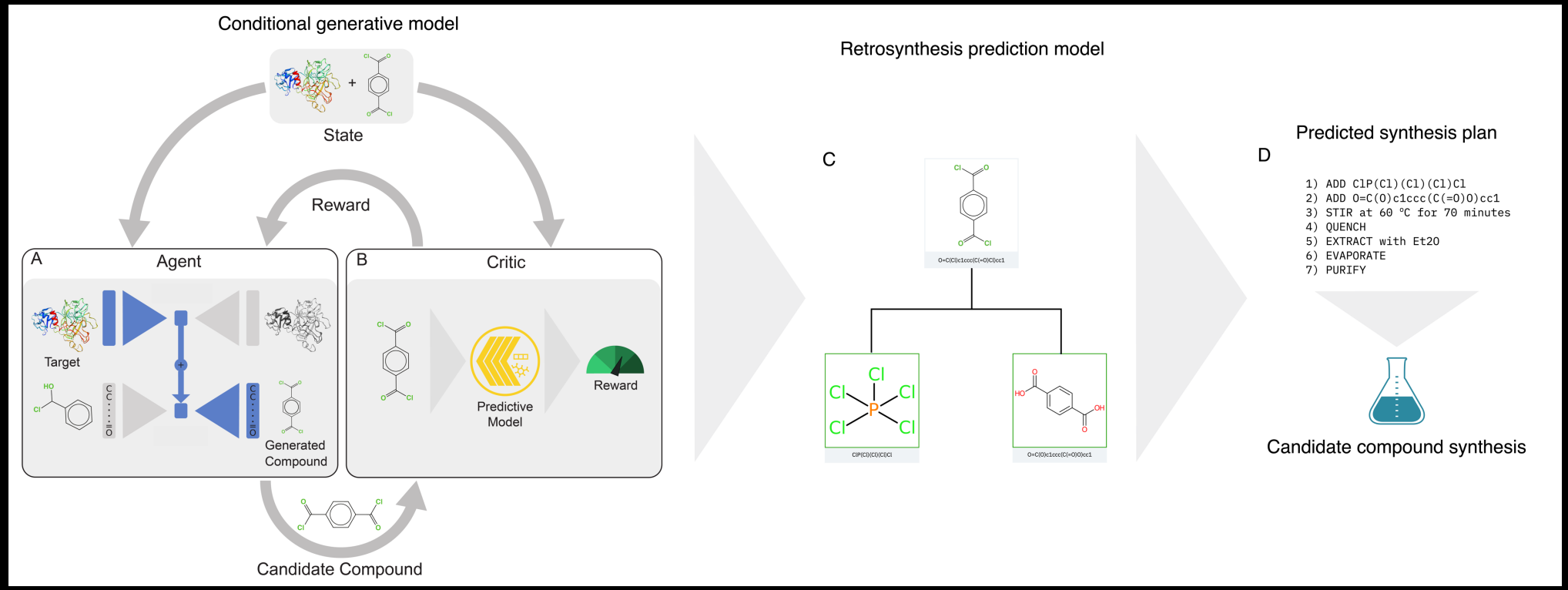
String-based (1D) Generative Molecule Creation - Gene Expression-driven Hit-like Molecule Design




```python
from gt4sd.algorithms.conditional_generation.paccmann_rl.core import (
    PaccMannRL, PaccMannRLOmicBasedGenerator
)
import numpy as np
# gene expression values normalized
target = np.array([0.5, ..., 0.9])
configuration = PaccMannRLOmicBasedGenerator()
algorithm = PaccMannRL(configuration=configuration, target=target)
molecules = list(algorithm.sample(15))
```

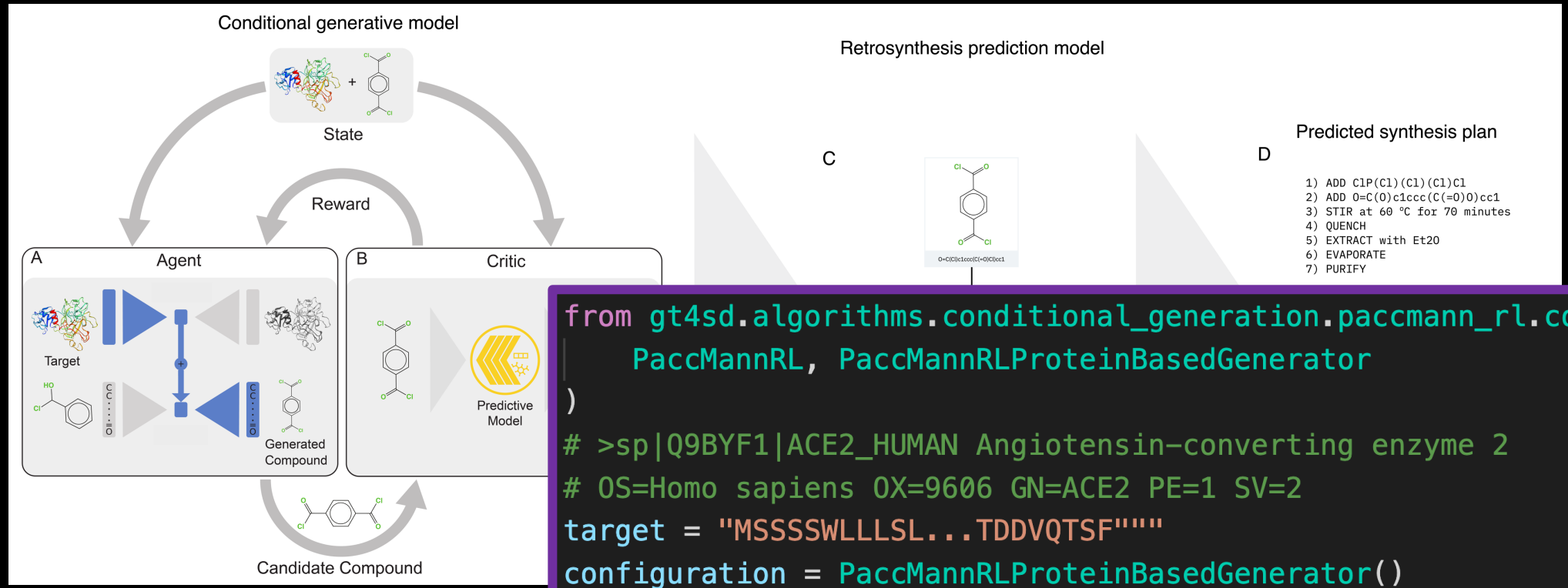
Born *et al.* (2021). PaccMannRL: De novo generation of hit-like anticancer molecules from transcriptomic data via reinforcement learning. *iScience*, 24, 4.

String-based (1D) Generative Molecule Creation - Data-driven Ligand Design and Synthesis



Born *et al.* (2021). Data-driven Molecular Design for Discovery and Synthesis of Novel Ligands-A case study on SARS-CoV-2. *MLST*, 2, 2.

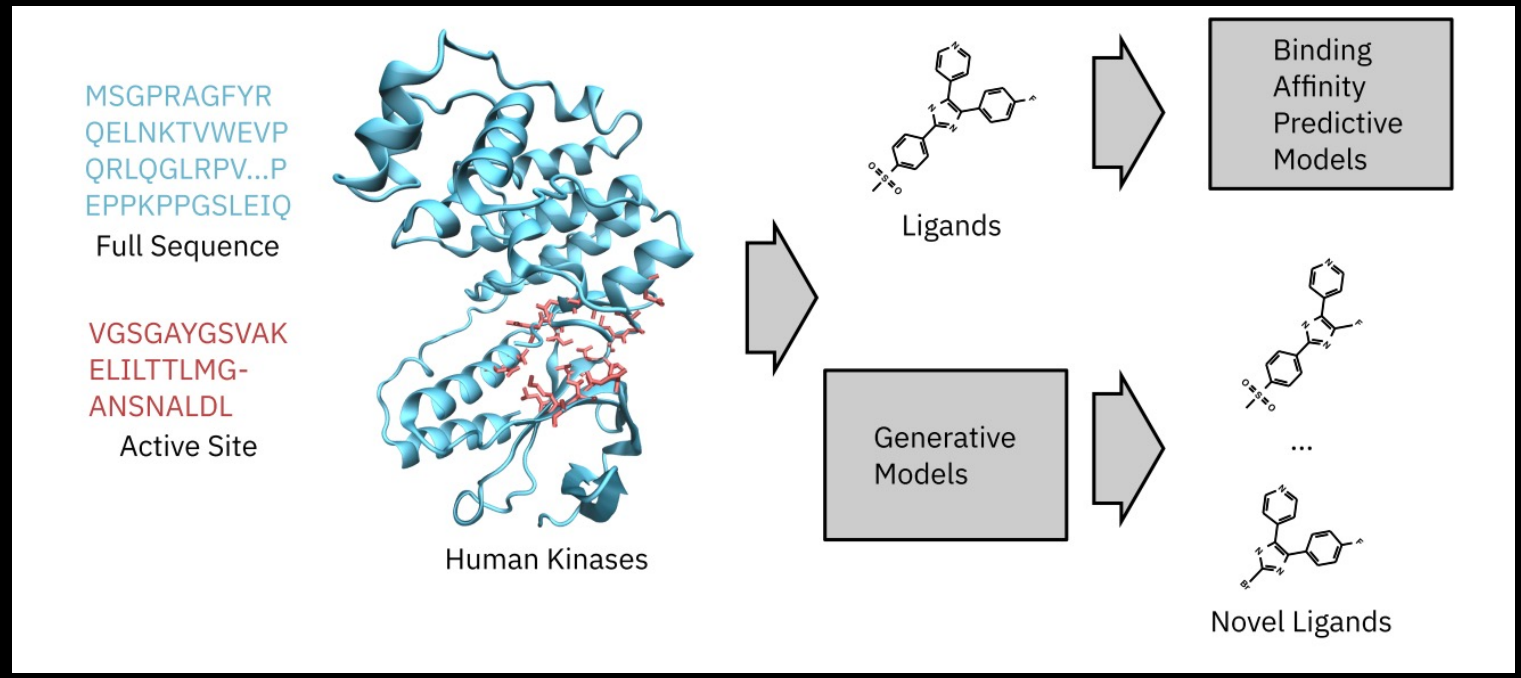
String-based (1D) Generative Molecule Creation - Data-driven Ligand Design and Synthesis

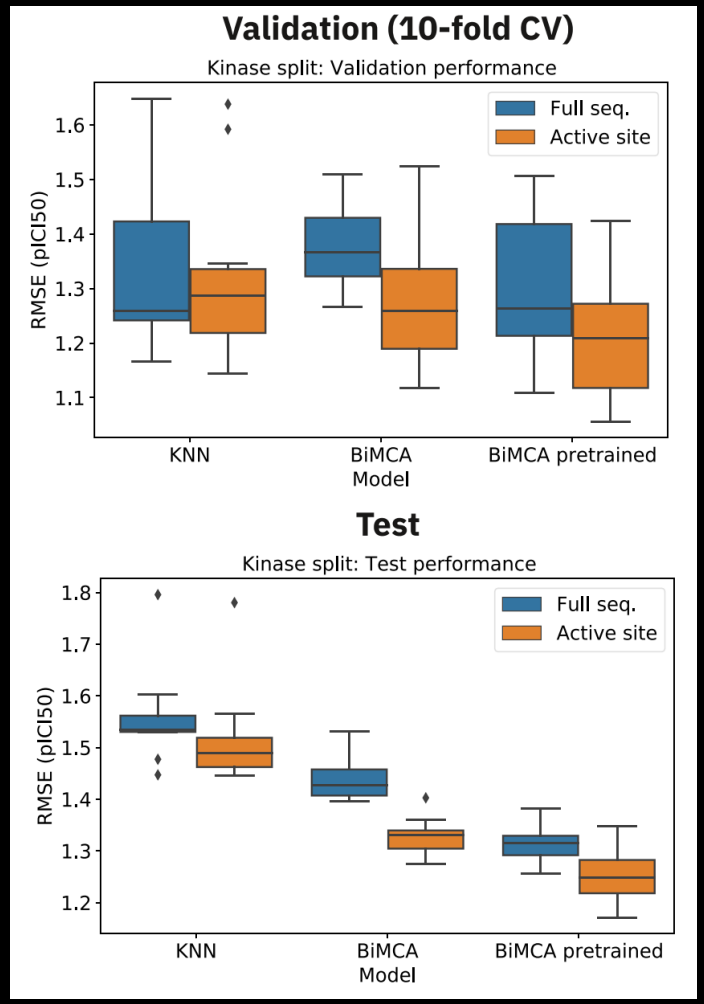



```python
from gt4sd.algorithms.conditional_generation.paccmann_rl.core import (
    PaccMannRL, PaccMannRLProteinBasedGenerator
)
# >sp|Q9BYF1|ACE2_HUMAN Angiotensin-converting enzyme 2
# OS=Homo sapiens OX=9606 GN=ACE2 PE=1 SV=2
target = "MSSSSWLLLSL...TDDVQTSF"""
configuration = PaccMannRLProteinBasedGenerator()
algorithm = PaccMannRL(configuration=configuration, target=target)
molecules = list(algorithm.sample(15))
```

Born *et al.* (2021). Data-driven Molecular Design for Discovery and Synthesis of Novel Ligands-A case study on SARS-CoV-2. *MLST*, 2, 2.

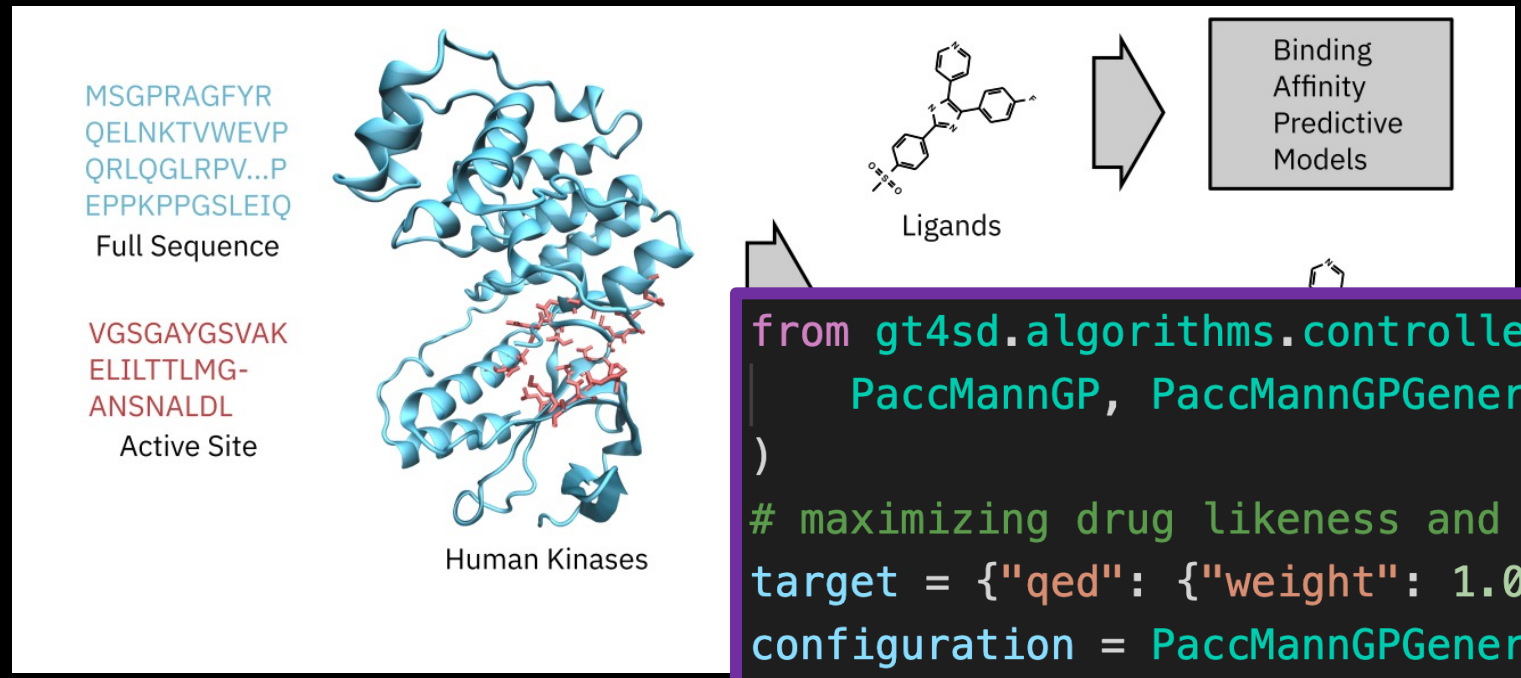
String-based (1D) Generative Molecule Creation - 3D Effects by Using Only Active Site Residues





Born *et al.* (2022). Active Site Sequence Representations of Human Kinases Outperform Full Sequence Representations for Affinity Prediction and Inhibitor Generation: 3D Effects in a 1D Model. *JCIM*, 62, 240-257.

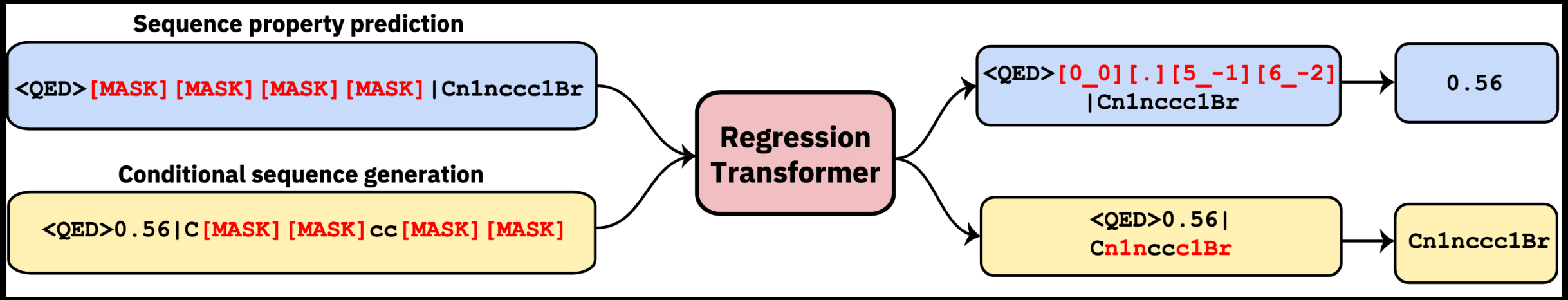
String-based (1D) Generative Molecule Creation - 3D Effects by Using Only Active Site Residues




```python
from gt4sd.algorithms.controlled_sampling.paccmann_gp.core import (
    PaccMannGP, PaccMannGPGenerator
)
# maximizing drug likeness and synthesizability
target = {"qed": {"weight": 1.0}, "sa": {"weight": 1.0}}
configuration = PaccMannGPGenerator()
algorithm = PaccMannGP(configuration=configuration, target=target)
molecules = list(algorithm.sample(15))
```

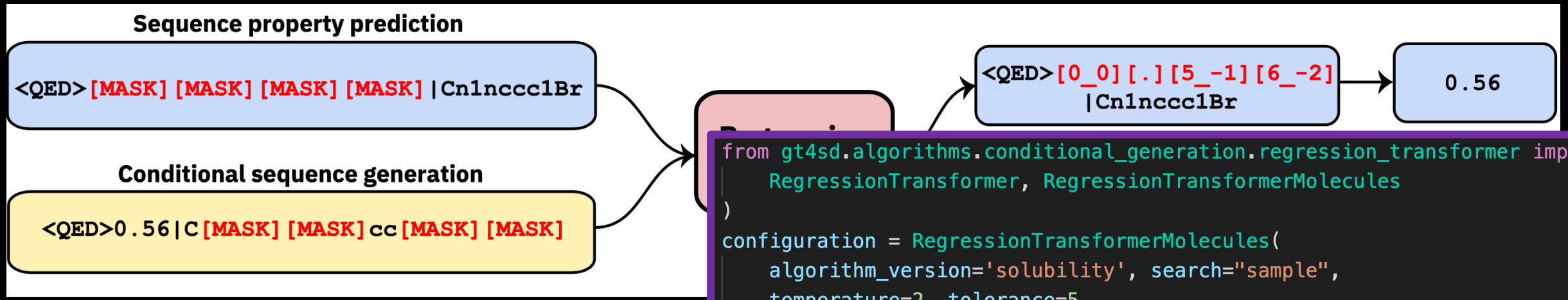
Born *et al.* (2022). Active Site Sequence Representations of Human Kinases Outperform Full Sequence Representations for Affinity Prediction and Inhibitor Generation: 3D Effects in a 1D Model. *JCIM*, 62, 240-257.

String-based (1D) Generative Molecule Creation - Multi-task Modeling for Conditional Generative Design



Born and Manica (2022). Regression Transformer: Concurrent Conditional Generation and Regression by Blending Numerical and Textual Tokens. *ICLR* Workshop Machine Learning for Drug Discovery

String-based (1D) Generative Molecule Creation - Multi-task Modeling for Conditional Generative Design




```python
from gt4sd.algorithms.conditional_generation.regression_transformer impo
    RegressionTransformer, RegressionTransformerMolecules
)
configuration = RegressionTransformerMolecules(
    algorithm_version='solubility', search="sample",
    temperature=2, tolerance=5
)
target = "<esol>-3.534|[C][C][MASK]..."
algorithm = RegressionTransformer(
    configuration=configuration, target=target
)
molecules = list(algorithm.sample(15))
```

Born and Manica (2022). Regression Transformer: Concurrent Conditional Generation and Regression by Blending Numerical and Textual Tokens. *ICLR* Workshop Machine Learning for Drug Discovery

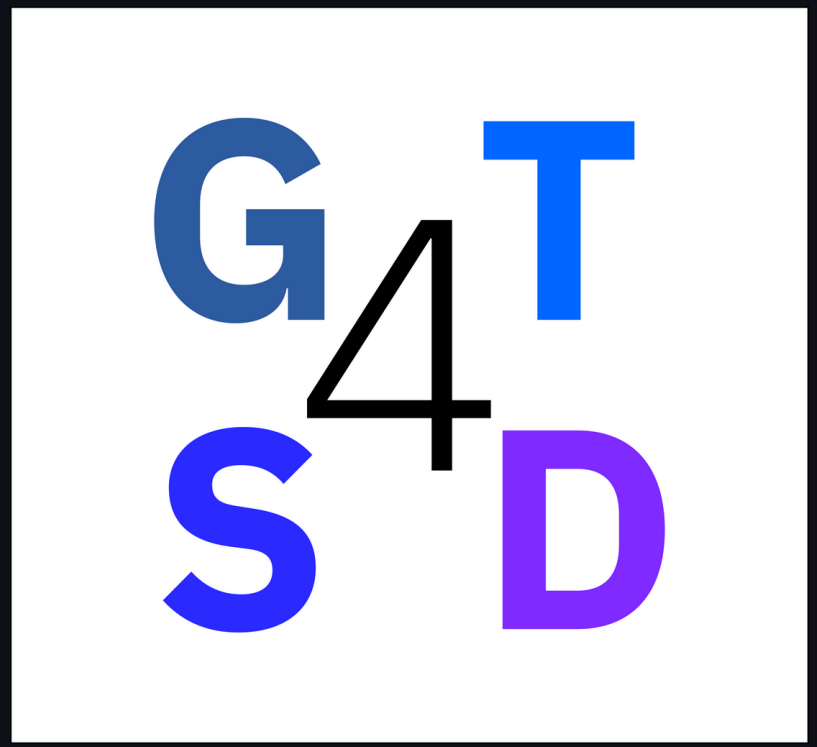
Thanks for your attention

GT4SD (Generative Toolkit for Scientific Discovery)

pypi package 0.44.0 | Running tests: style, pytests and entry-points passing | License MIT | code style black | contributions welcome

website live | downloads 10k | downloads/month 3k | launch binder | DOI 10.5281/zenodo.6798761

Award 2022 IEEE Open Software Services Award



If you want to know more check out the available resources:

GT4SD/gt4sd-core

Pre-print:2207.03928

Documentation

Contributions are welcome:

CONTRIBUTING.md

CODE_OF_CONDUCT.md