PyTDC: A multimodal machine learning training, evaluation, and inference platform for biomedical foundation models

Alejandro Velez-Arce [1 2] **Marinka Zitnik** [3 4 5 6]

Abstract

Existing biomedical benchmarks do not provide end-to-end infrastructure for training, evaluation, and inference of models that integrate multimodal biological data and a broad range of machine learning tasks in therapeutics. We present PyTDC, an open-source machine-learning platform providing streamlined training, evaluation, and inference software for multimodal biological AI models. PyTDC unifies distributed, heterogeneous, continuously updated data sources and model weights and standardizes benchmarking and inference endpoints. This paper discusses the components of PyTDC's architecture and, to our knowledge, the first-of-its-kind case study on the introduced single-cell drug-target nomination ML task. We find state-of-the-art methods in graph representation learning and domain-specific methods from graph theory perform poorly on this task. Though we find a context-aware geometric deep learning method that outperforms the evaluated SoTA and domain-specific baseline methods, the model is unable to generalize to unseen cell types or incorporate additional modalities, highlighting PyTDC's capacity to facilitate an exciting avenue of research developing multimodal, context-aware, foundation models for open problems in biomedical AI.

[1]ArcellAI Inc. [2]calculus.house, The Residency Inc. [3]Department of Biomedical Informatics, Harvard Medical School, Boston, MA, USA [4]Broad Institute of MIT and Harvard, Cambridge, MA, USA [5]Harvard Data Science Initiative, Harvard University, Cambridge, MA, USA [6]Kempner Institute for the Study of Natural and Artificial Intelligence, Harvard University, Cambridge, MA, USA. Correspondence to: Alejandro Velez-Arce <alejandro.velez.arce@@gmail.com>, Marinka Zitnik <marinka@hms.harvard.edu>.

Proceedings of the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

1. Introduction

The application of transfer learning in biology has gained momentum, marking a pivotal shift toward harnessing the wealth of biological measurements available today. Foundation models pre-trained on structural, genomic, and network biology data have demonstrated remarkable potential in improving predictions in data-limited scenarios and accelerating biological discoveries (Theodoris, 2024). These models capture fundamental patterns in biological systems, creating a foundation for tasks ranging from disease modeling to therapeutic development. As this field advances, the need for robust strategies to benchmark these models becomes increasingly important. (Liu et al., 2023) illustrate the importance of diverse downstream task evaluations in assessing the utility of network biology foundation models. However, to be useful in therapeutics, benchmarking must also align with biomedical goals. Focusing benchmarks solely on canonical biology tasks such as cell type clustering and annotation neglects deeper biological insights, such as understanding disease mechanisms. Therefore, developing benchmarks prioritizing therapeutic relevance is essential to guide the evolution of models and ensure they address the challenges at the heart of biomedical research.

Single-cell genomics has enabled the study of cellular processes with remarkable resolution, offering insights into cellular heterogeneity and dynamics (Luecken et al., 2023). Progress in data generation and computational methods designed for single-cell analysis (CZI Single-Cell Biology, et al., 2023) has facilitated context-aware machine learning models that incorporate cell-type-specific data across various therapeutic areas (Chen et al., 2024). Despite these advances, there remains a need for datasets, benchmarks, and tools for reproducible development and benchmarking of out-of-distribution generalization (OOD) (Ektefaie et al., 2024) and domain-specific metric (Velez-Arce et al., 2024) performance of methods that integrate single-cell analysis with diverse therapeutic approaches. Given key problems in genomics intrinsically involve multiple modalities and adapting general-purpose sequence models for these tasks remains unclear (Garau-Luis et al., 2024), any system facilitating method development in single-cell therapeutics must incorporate numerous biomedical modalities.

Kahn et al. (2022) shows us that when domain-specific demands on machine learning solutions increase, framework innovation is often the catalyst for accelerated scientific research by prioritizing open, modular, customizable internals and state-of-the-art, research-ready models and training setups across a variety of domains. As such, we've developed the first (table 1) platform facilitating the curation of multimodal single-cell data, retrieval and inference of context-aware biomedical representation learning models, and therapeutic-task-specific method fine-tuning and benchmarking. We present PyTDC (figure 1), a machine-learning platform providing streamlined training, evaluation, and inference software for single- cell biological foundation models. PyTDC introduces an API-first architecture (Beaulieu et al., 2022) that unifies heterogeneous, continuously updated data sources. The platform introduces a model server (figure 2), which provides unified access to continuously updated model weights across distributed repositories and standardized inference endpoints. Building upon Therapeutic Data Commons (Huang et al., 2021; 2022; Velez-Arce et al., 2024), we present single-cell therapeutics tasks, datasets, and benchmarks for model development and evaluation on OOD predictions and domain-specific metrics.

Overall, the contributions can be summarized as (highlighted in Table 1):

1. **We integrate single-cell analysis with multimodal machine learning in therapeutics via the introduction of contextualized tasks.** PyTDC presents datasets, models, and benchmarks for: single-cell drug-target nomination (Li et al., 2024; Velez-Arce et al., 2024) (section 4, 4.3, and A.1), single-cell chemical and genetic perturbation response predictions (Hetzel et al., 2022; Roohani et al., 2023) (section A.2), and cell-type-specific protein-peptide interaction prediction (Grazioli et al., 2022a; Brown et al., 2023) (section A.3). In our benchmarks we measure: the ability of models perform at most predictive biological contexts (Kwon et al., 2024), perturbation response prediction performance at out-of-distribution samples (Piran et al., 2023), and model robustness to negative sampling heuristics for out-of-distribution predictions (Kim et al., 2023).

2. **We introduce a collection of continually updated heterogeneous biomedical data sources enabled via a novel architecture.** PyTDC's modalities span but are not limited to: single-cell gene expression atlases (CZI Single-Cell Biology, et al., 2023; Jones et al., 2022), single-cell chemical (Aissa et al., 2021; Klein, 2023) and genetic (Norman et al., 2019; Replogle et al., 2022) perturbations (Peidli et al., 2024), clinical trial outcome data (Fu et al., 2022), peptide sequence data (Grazioli et al., 2022b; Gao et al., 2023), peptidomimetics protein-peptide interaction data from AS-MS spectroscopy (Brown et al., 2023; Ye et al., 2022), 3D structural protein data (Meslamani et al., 2011; Mysinger et al., 2012; Liu et al., 2015), cell-type-specific protein-protein interaction networks at single-cell resolution (Li et al., 2024), and biomedical knowledge graphs (Chandak et al., 2023). The novel API-first (Beaulieu et al., 2022) (section 3.1) architecture provides abstractions for complex data processing pipelines (Huang et al., 2021), data lineage (Thiago et al., 2020), and versioning (van der Weide et al., 2017). It augments the stability of emerging biomedical AI workflows, based on advancements from (Schick et al., 2023; Patil et al., 2023; M. Bran et al., 2024), with continuous data updates (Liu et al., 2022). More details in section 3.1 and B.1.

3. **We have released open source model retrieval and deployment software that streamlines AI inferencing, downstream fine-tuning, and domain-specific evaluation for representation learning models across biomedical modalities.** The PyTDC model server (sections 3.2 and C and figures 2 and 1) facilitates the use and evaluation of biomedical foundation models for downstream therapeutic tasks and enforces alignment between new method development and therapeutic objectives. We have released support for inference on several state-of-the-art (SoTA) models (Lopez et al., 2018; Theodoris et al., 2023; Cui et al., 2024; Li et al., 2024; ESM Team, 2024) (section C) as well as benchmarking software for domain-specific evaluation (Velez-Arce et al., 2024).

In this work (section 4), we present a case study on single-cell drug-target nomination (Li et al., 2024; Velez-Arce et al., 2024). PyTDC facilitates our analysis showing inadequate performance of domain-specific methods and state-of-the-art (SoTA) general-purpose graph representation learning methods. Using a novel and adaptable framework for contextualized metrics (sections 3.3 and 4.2), we measure therapeutic relevance of these methods by their predictions at most predictive cell types (Zhang et al., 2022; Kwon et al., 2024) and find superior performance by (Li et al., 2024) on these domain-specific metrics. Though a promising result, the inability of (Li et al., 2024) to generalize to unseen cell types and incorporate additional modalities leaves this as an exciting machine learning research task in therapeutics for the PyTDC-enabled development of methods leveraging foundation models and transfer learning.

2. Related Work

Datasets, benchmarks, and platforms. Numerous datasets, benchmarks, and platforms have been released supporting training and evaluation of predictive models on therapeutic tasks (Wu et al., 2018; Huang et al., 2021; Xu et al., 2022; Huang et al., 2022; Zhu et al., 2022; Li et al., 2022).

Table 1. **Assessment of SoTA datasets, benchmarks, and ML platforms** Focusing on single-cell therapeutics, data modalities, and model inferencing support, we compare PyTDC's capabilities, including features developed by the (Velez-Arce et al., 2024) team, with the SoTA platforms (Wu et al., 2018; Huang et al., 2021; Zhu et al., 2022; CZI Single-Cell Biology, et al., 2023; Luecken et al., 2023; Peidli et al., 2024). We find PyTDC outperforms in its coverage of single-cell therapeutic tasks (sections 4 and A) and data modalities (sections 3.1 and B) and in its support of novel inferencing capabilities for representation learning models (sections 3.2 and C) and domain-specific (section 3.3) evaluation of transfer learning methods.

Feature	TDC	TorchDrug	MoleculeNet	OpenProblems	scPerturb	CELLXGENE	PyTDC
Single-cell Drug-Target Identification Task	✗	✗	✗	✗	✗	✗	✓
CRISPR-Cell Perturbation Response Prediction	✗	✗	✗	✗	✓	✗	✓
Drug-Cell Perturbation Response Prediction	✗	✗	✗	✗	✓	✗	✓
Single-cell Protein-Peptide Binding Interaction Prediction	✗	✗	✗	✗	✗	✗	✓
Structure-Based Drug Design	✗	✓	✓	✗	✗	✗	✓
Clinical Trial Outcome Prediction	✗	✗	✗	✗	✗	✗	✓
Knowledge Graphs	✗	✓	✗	✗	✗	✗	✓
Python API	✓	✓	✗	✓	✗	✓	✓
Single-cell Gene Expression Atlases	✗	✗	✗	✓	✓	✓	✓
External API data retrieval	✗	✗	✗	✗	✗	✗	✓
Data Views	✗	✗	✗	✗	✗	✗	✓
Pre-computed representation learning model embedding retrieval	✗	✗	✗	✗	✗	✓	✓
AI Inferencing for pre-trained representation learning models	✗	✓	✗	✗	✗	✗	✓
AI Inferencing for pre-trained transformer models	✗	✗	✗	✗	✗	✗	✓

There have also been more recent releases of benchmarks for canonical single-cell biology ML tasks (CZI Single-Cell Biology, et al., 2023; Luecken et al., 2023), including of representation learning models in single-cell biology (Initiative, 2023). None of these systems are tailored to meaningful evaluation of representation learning methods (Lopez et al., 2018; Cui et al., 2024; Theodoris et al., 2023; Chen et al., 2024; Li et al., 2024) at therapeutic tasks (Velez-Arce et al., 2024). PyTDC has been developed with the datasets, architecture, and model training, evaluation, and inference capabilities detailed in table 1, figures 2 and 1, and section 3, to address the need for a multimodal machine learning platform for the development of biomedical methods leveraging transfer learning (Theodoris, 2024).

Therapeutic and single-cell foundation models. Foundation models trained on Therapeutics Data Commons (Huang et al., 2021; 2022; Velez-Arce et al., 2024) have been shown to generalize across several therapeutic tasks (Zambrano Chaves et al., 2024). Additionally, parallel efforts in training foundation models on large single-cell atlases have shown a potential to advance cell type annotation and matching of healthy-disease cells to study cellular signatures of disease (Yang et al., 2022; Hao et al., 2024; Cui et al., 2024). PyTDC bridges these efforts by providing formal definitions of therapeutic tasks, datasets, and benchmarks at single-cell resolution in order to provide precise therapeutic predictions incorporating cell type contexts (Kwon et al., 2024; Li et al., 2024) and developing machine learning infrastructure for streamlining transfer learning method (Theodoris, 2024) development, evaluation, and inference.

3. Streamlining training, evaluation, and inference

3.1. Multimodal Single-cell Data Retrieval

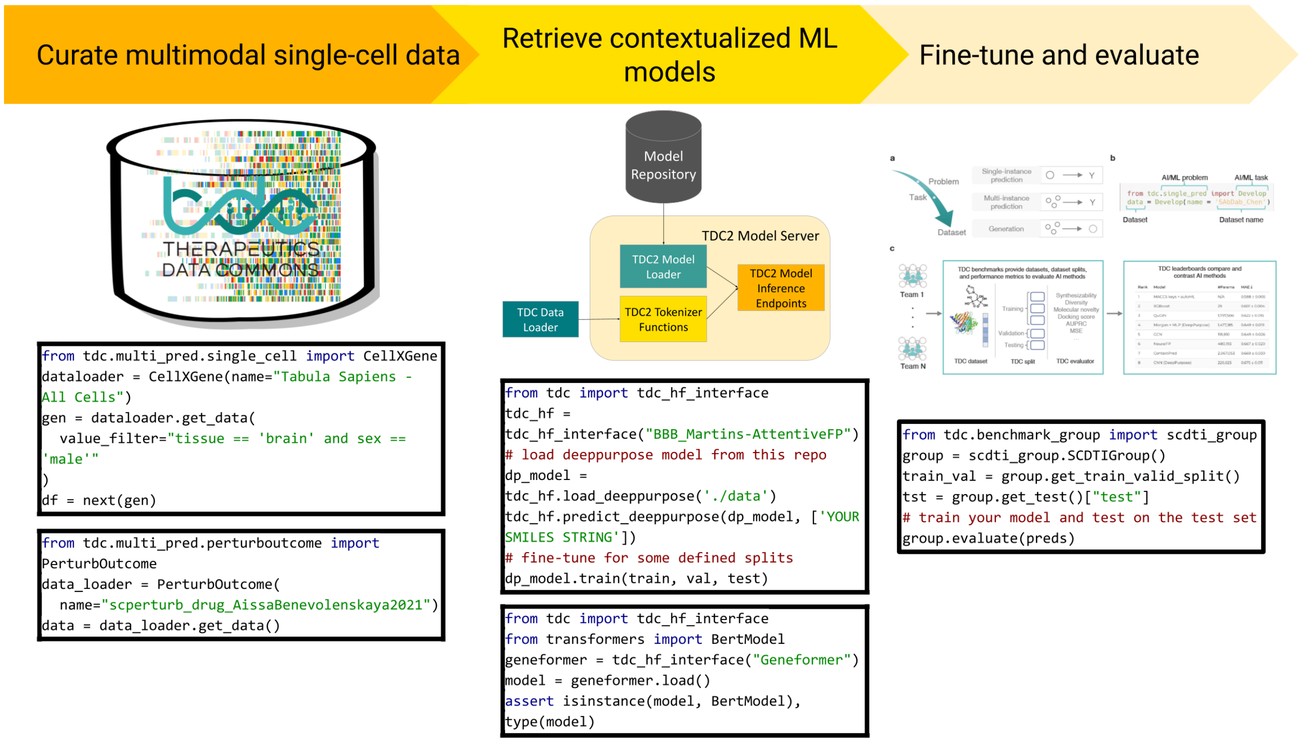
A collection of multimodal continually updated heterogeneous data sources is unified under the introduced "API-first-dataset" architecture. Inspired by API-first design (Beaulieu et al., 2022), this microservice architecture is implemented using the model-view-controller design pattern (Bucanek, 2009; Malik et al., 2021) to enable multimodal data views (Churi et al., 2016; Emmerik et al., 1993; Rammerstorfer & Mössenböck, 2003) under a domain-specific-language (Membarth et al., 2016). We discuss further details and data sources for the modalities in section 1 and tasks in table 1.

3.2. PyTDC Model Server

PyTDC presents open source model retrieval and deployment software that streamlines AI inferencing and exposes state-of-the-art, research-ready models and training setups for biomedical representation learning models across modalities. In section C and figure 2, we describe the architectural components of the model server and its integration into the model development lifecycle with the TDC platform. In figure 1, we show the components of PyTDC enabling the full development lifecycle for transfer learning ML methods in therapeutics. Figure 3 also shares how a workflow spanning hundreds-to-thousands of lines of code is simplified to below 30 with PyTDC's model server and data retrieval capabilities.

3.3. Context-specific metrics

In machine learning applications, data subsets can correspond to critical outcomes. In therapeutics, there is evidence



```
from tdc.multi_pred.single_cell import CellXGene
dataloader = CellXGene(name="Tabula Sapiens -
All Cells")
gen = dataloader.get_data(
  value_filter="tissue == 'brain' and sex ==
'male'"
)
df = next(gen)
```

```
from tdc.multi_pred.perturboutcome import
PerturbOutcome
data_loader = PerturbOutcome(
  name="scperturb_drug_AissaBenevolenskaya2021")
data = data_loader.get_data()
```

```
from tdc import tdc_hf_interface
tdc_hf =
tdc_hf_interface("BBB_Martins-AttentiveFP")
# load deeppurpose model from this repo
dp_model =
tdc_hf.load_deeppurpose('./data')
tdc_hf.predict_deeppurpose(dp_model, ['YOUR
SMILES STRING'])
# fine-tune for some defined splits
dp_model.train(train, val, test)
```

```
from tdc import tdc_hf_interface
from transformers import BertModel
geneformer = tdc_hf_interface("Geneformer")
model = geneformer.load()
assert isinstance(model, BertModel),
type(model)
```

```
from tdc.benchmark_group import scdti_group
group = scdti_group.SCDTIGroup()
train_val = group.get_train_valid_split()
tst = group.get_test()["test"]
# train your model and test on the test set
group.evaluate(preds)
```

Figure 1. We present PyTDC, a machine-learning platform providing streamlined training, evaluation, and inference software for single-cell biological foundation models to accelerate research in transfer learning method development in therapeutics (Theodoris, 2024). PyTDC introduces an API-first (Beaulieu et al., 2022) architecture (sections 3.1 and B) that unifies heterogeneous, continuously updated data sources. The platform introduces a model server, which provides unified access to model weights across distributed repositories and standardized inference endpoints (sections 3.2, B.3.3, and C). The model server accelerates research workflows (Kahn et al., 2022) by exposing state-of-the-art, research-ready models and training setups for biomedical representation learning models across modalities. Building upon Therapeutic Data Commons (Huang et al., 2021; 2022; Velez-Arce et al., 2024), we present single-cell therapeutics tasks, datasets, and benchmarks for model development and evaluation (sections 4 and A).

that the effects of drugs can vary depending on the type of cell they are targeting and where specific proteins are acting (Zhang et al., 2022). Inspired by the success of candidate Graph Foundation Model architectures (Huang et al., 2023) at subgraph-based views for classification tasks (Mao et al., 2024) and building upon the concept of 'slices' (Chen et al., 2020a), we assess model performance on critical biological subsets by introducing the framework and definitions for contextualized metrics to measure model performance at critical biological slices. Our benchmark for single-cell drug-target nomination measures model performance at cell-type-specific subgraphs by computing classification and retrieval metrics aggregated over the top-performing cell types. See Section 4.2 for definitions.

4. Evaluation of graph representation learning models at context-aware drug-target nomination

Identifying disease pathways, which are groups of proteins linked to specific diseases, is a crucial area of research that can lead to valuable insights for diagnosing, prognosis, and treating diseases. Computational methods facilitate this discovery process by utilizing protein-protein interaction (PPI) networks. These methods typically begin with a few known disease-associated proteins and seek to identify additional proteins in the pathway by exploring the PPI network surrounding these known proteins. However, these approaches have often had limited success, and the reasons for their failures are poorly understood (Agrawal et al., 2018). Single-cell analysis provides a promising pathway to address this

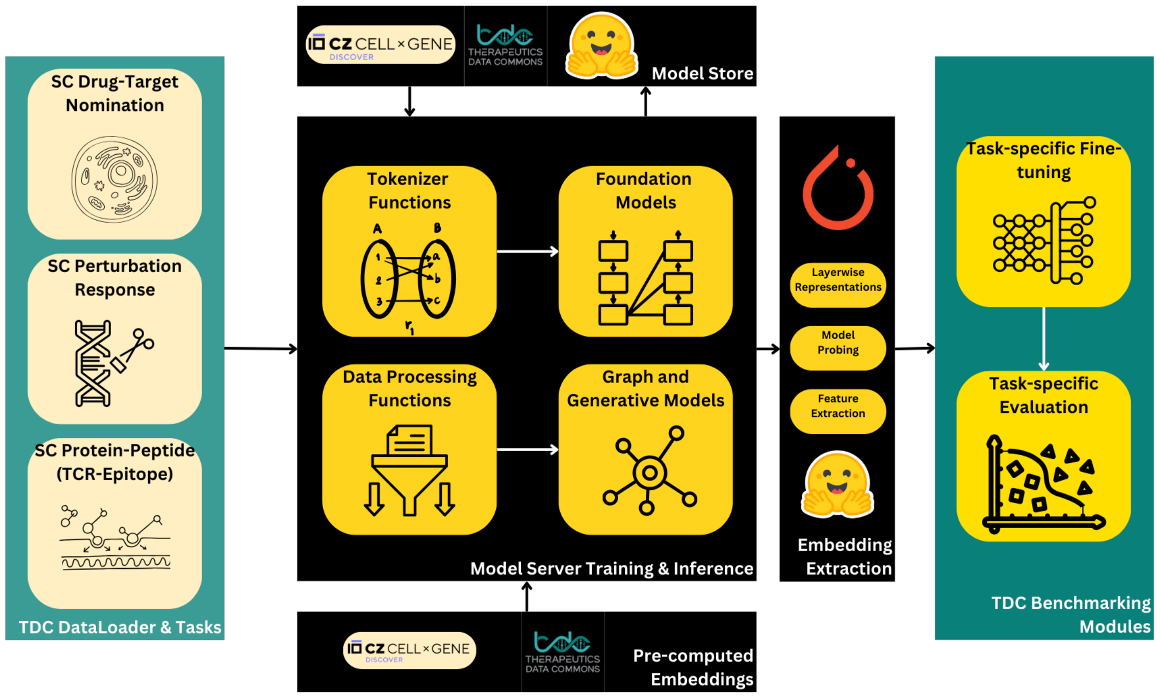


Figure 2. **AI inferencing and model evaluation components.** The PyTDC model server (sections 3.2 and C) streamlines retrieval, inferencing, and training setup for an array of context-aware biological foundation models and models spanning multiple modalities. A model store retrieval API provides unified access to model weights stored in the Hugging Face Model Hub (https://huggingface.co/apliko), Chan-Zuckerberg CELLxGENE Census fine-tuned models, and TDC (Huang et al., 2021; 2022; Velez-Arce et al., 2024) storage. The model server also provides access to model classes, tokenizer functions, and inference endpoints supporting PyTorch (Paszke et al., 2019) and Hugging Face Transformers (Wolf et al., 2020). Extracted embeddings, from either model server inference or pre-computed embedding storage, are ready for downstream use by task-specific benchmarking modules.

by enabling the study of gene expression and function at the level of individual cells across healthy and disease states (Jones et al., 2022; CZI Single-Cell Biology, et al., 2023; Kwon et al., 2024). To facilitate biological discoveries using single-cell data, machine-learning models have been developed to capture the complex, cell-type-specific behavior of genes (Lopez et al., 2018; Yang et al., 2022; Theodoris et al., 2023; Cui et al., 2024; Li et al., 2024). PyTDC introduces a task, dataset, and benchmark for the development of contextual AI models to nominate therapeutic targets in a cell type-specific manner (Li et al., 2024).

Machine Learning Challenges. Fragmentation of disease and drug pathways when overlaid on networks has been studied quite extensively. In fact, an overwhelming majority of pathways don't correspond to well-connected components in PPI networks, leading to poor performance of state-of-the-art disease pathway discovery methods (Agrawal et al., 2018). One promising technique in machine learning for omics is identifying the most predictive cell-type contexts for a biomarker (Velez-Arce et al., 2024). This approach can help determine the cell types that play crucial and distinct roles in the disease pathogenesis of conditions like rheumatoid arthritis (RA) and inflammatory bowel diseases (IBD) (Li et al., 2024). To our knowledge, no work systematically evaluates state-of-the-art methods' ability to identify most predictive cell types in drug target nomination.

4.1. Single-cell drug-target nomination formulation

Single-cell drug-target nomination (`tdc_ml.scDTN`) is a predictive, non-generative task formalized as learning an estimator for a disease-specific function f of a target protein and cell type outputting whether the candidate protein t is a therapeutic target in that cell type c:

$$y = f(t, c). \tag{1}$$

Target candidate set. The target candidate set includes proteins, nucleic acids, or other molecules drugs can interact with, producing a therapeutic effect or causing a biological response. The target candidate set is constrained to proteins relevant to the disease being treated. It is denoted by:

$$\mathbb{T} = \{t_1, \ldots, t_{N_t}\}, \tag{2}$$

Figure 3. The below code illustrates the integration of PyTDC's single-cell datasets with the model server (sections 3.2, C, and B.3.3). Here we retrieve the (Peidli et al., 2024; Aissa et al., 2021) chemical perturbation response dataset (section A.2.4, A.2, and A.2.2) and extract gene expression vector embeddings from (Theodoris et al., 2023). Such a workflow would take a user hundreds-to-thousands of lines of code to develop. PyTDC allows the user to extract single-cell foundation model embeddings from complex and customized gene expression datasets with less than 30 lines of code.

```
from tdc_ml.multi_pred.perturboutcome import PerturbOutcome
from tdc_ml import tdc_hf_interface
import torch

dataset = "scperturb_drug_AissaBenevolenskaya2021"
data = PerturbOutcome(dataset) # import dataset for chemical perturbation response
    prediction
adata = data.adata
tokenizer = GeneformerTokenizer(max_input_size=3) # initialize model tokenizer
adata.var["feature_id"] = adata.var.index.map(
    lambda x: tokenizer.gene_name_id_dict.get(x, 0)) # format features using tokenizer
        data processing functions
x = tokenizer.tokenize_cell_vectors(adata, ensembl_id="feature_id", ncounts="counts") #
    tokenize custom dataset
cells, _ = x
geneformer = tdc_hf_interface("Geneformer")
model = geneformer.load() # load the Geneformer model
input_tensor = torch.tensor(cells)
# note you'd typically batch the input tensor
attention_mask = torch.tensor([[t != 0 for t in cell] for cell in batch])
model(batch,
    attention_mask=attention_mask,
    output_hidden_states=True)
```

where $t_1, \ldots, t_{N_t}$ are N_t target candidates for the drugs treating the disease. Information modeled for target candidates can include interaction, structural, and sequence information.

Biological context set. The biological context set includes the cell-type-specific contexts in which the target candidate set operates. This set is denoted as:

$$\mathbb{C} = \{c_1, \ldots, c_{N_c}\}, \tag{3}$$

where $c_1, \ldots, c_{N_c}$ are N_c biological contexts on which drug-target interactions are being evaluated. Information modeled for cell-type-specific biological contexts can include gene expression and tissue hierarchy. The set is constrained to disease-specific cell types and tissues.

Drug-target nomination. Drug-Target Nomination is a binary label $y \in \{1, 0\}$, where $y = 1$ indicates the protein is a candidate therapeutic target. At the same time, 0 means the protein is not such a target.

The goal is to train a model f_θ for predicting the probability $\hat{y} \in [0, 1]$ that a protein is a candidate therapeutic target in a specific cell type. The model learns an estimator for a disease-specific function of a protein target $t \in \mathbb{T}$ and a cell-type-specific biological context $c \in \mathbb{C}$ as input, and the model is tasked to predict:

$$\hat{y} = f_\theta(t \in \mathbb{T}, c \in \mathbb{C}). \tag{4}$$

4.2. Context-Specific Metrics in `tdc_ml.scDTN`

Context-specific metrics (see section 3.3) are defined to measure model performance at critical biological slices, with our benchmarks focused on measuring cell-type-specific model performance. For single-cell drug-target nomination, we measure model performance at top-performing cell types. We evaluate predictive cell type retrieval using average precision at rank R (AP@R) and target classification using AUROC, both at top K predicted cell types. Formally, we define context-specific AP@R and AUROC below.

Context-specific AUROC. To calculate the AUROC for the top K performing cell types, we first need to determine which cell types achieve the highest AUROC scores. After selecting the top-performing cell types, we weigh each top-performing cell type's AUROC score by the number of samples in that cell type.

We denote:

$$\mathbb{D} = \{(x_i, y_i, c_i)\}, \quad \forall i \in \mathbb{S} \tag{5}$$

Here, $\mathbb{D}$ denotes the dataset where x_i denotes the feature vector, y_i is the true label, and c_i is the cell type for sample i from $\mathbb{S}$. We further denote C, the set of unique cell types. Then, the AUROC for a specific cell type, $AUROC_c$, is computed as:

$$AUROC_c = AUROC(D_c) \tag{6}$$

Here, $D_c = \{(x_i, y_i) | c_i = c\}$ is the subset of the dataset for cell type c and $AUROC(D_c)$ represents the AUROC score computed over this subset. Once these are computed, values can be sorted in descending order to select the top X cell type with highest AUROC value.

$$\begin{aligned} C_K = \{c_1, c_2, \ldots, c_K\} \quad \text{s.t.} \\ AUROC_{c_i} \geq AUROC_{c_j}, \\ \forall i \leq K, j > K \end{aligned} \tag{7}$$

The weighted AUROC for the top K cell types is given by weighting each cell type's AUROC by the proportion of its samples relative to the total samples in the top K cell types.

$$AUROC_{TopK} = \frac{\sum_{c \in C_K} AUROC_c \times |D_c|}{\sum_{c \in C_K} |D_c|} \tag{8}$$

This measure represents a balance between representation and performance of the cell types, and primarily evaluates the model's classification strengths (Velez-Arce et al., 2024).

Context-specific Average Precision at rank R (AP@R). In our study, we compute AP@R for the top K performing cell types. We denote dataset and samples as above.

$$\mathbb{D} = \{(x_i, y_i, c_i)\}, \quad \forall i \in \mathbb{S} \tag{9}$$

Here, $\mathbb{D}$ denotes the dataset where x_i denotes the feature vector, y_i is the true label, and c_i is the cell type for sample i from $\mathbb{S}$. We further denote C, the set of unique cell types. The samples of each cell type, $D_c = (x_i, y_i) | c_i = c$, can be sorted based on the score output by the model for said sample $f(x_i)$, with average precision at rank type computed accordingly.

$$\begin{aligned} D_c^R = \{x_1, \ldots, x_R\} \quad \text{s.t.} \\ f(x_i) \geq f(x_j), \\ \forall i \leq R, j > R, c_i = c, c_j = c \end{aligned} \tag{10}$$

$$AP@R_c = AP(\{y_1, \ldots, y_R\}, \{f(x_1), \ldots, f(x_R)\}), \quad x_i \in D_c^R \tag{11}$$

The average precision at rank R at Top K cell types can then be defined as:

$$\begin{aligned} C_K = \{c_1, c_2, \ldots, c_K\} \quad \text{s.t.} \\ AP@R_{c_i} \geq AP@5_{c_j}, \\ \forall i \leq K, j > K \end{aligned} \tag{12}$$

$$AP@R_{TopK} = mean(\{AP@R_{c_i}\}, \quad \forall c_i \in C_K) \tag{13}$$

AP@R is robust to varied sizes among cell-type-specific subgraphs and number of protein targets in those subgraphs (Li et al., 2024; Velez-Arce et al., 2024) and it is used as the retrieval evaluation metric.

4.3. Drug-target Nomination Benchmarks

We use curated therapeutic target labels from the Open Targets Platform (Targets, 2023) for rheumatoid arthritis (RA) and inflammatory bowel disease (IBD) (Li et al., 2024) (section A.1.2). Previous work (Velez-Arce et al., 2024; Li et al., 2024) benchmarked PINNACLE (Li et al., 2024)—trained on cell type specific protein-protein interaction networks—and a graph attention neural network (GAT) (Velickovic et al., 2017)—trained on a context-free reference protein-protein interaction network—on the curated therapeutic targets dataset. The results (Appendix tables 4 and 3) showed PINNACLE outperforming GAT when it's performance on contextualized metrics (section 4.2) was compared to GAT's aggregate, context-free, performance. However, this evaluation does not establish the difficulty of performing well on contextualized metrics for drug-target nomination or motivate the need for stronger machine learning models for this task. To our knowledge, PyTDC is the first to make this case, evaluate state-of-the-art methods at the task, and streamline training, evaluation, and inference in single-cell drug-target nomination.

As a domain-specific baseline, we run a label propagation algorithm Aridhi et al. (2017) over the Li et al. (2024) cell type specific PPI by labeling the train and validation sets and propagating to the unlabeled test set via a majority vote among the neighborhood of an unlabeled node. We find a significant portion of the graph remains unlabeled after convergence, resulting in poor performance of the domain-specific baseline illustrated in table 2. We attribute this result to the fragmentation of RA and IBD pathways when overlaid on PPI networks (Agrawal et al., 2018).

For comparison, we evaluate a state-of-the-art architecture for representation learning with graphs, Brody et al. (2022), which incorporates an attention function. In addition, we evaluate an attention-free architecture for representation learning with graphs, Grover & Leskovec (2016). In both cases, we one-hot encode cell type labels to the context-free PPI graph used to evaluate GAT in Velez-Arce et al. (2024); Li et al. (2024).

Table 2. **Cell-type specific target nomination for two therapeutic areas, rheumatoid arthritis (RA) and inflammatory bowel diseases (IBD).** Cell-type specific context metrics (see 3.3 and definitions in Section 4.2) are used to benchmark (Grover & Leskovec, 2016; Aridhi et al., 2017; Brody et al., 2022; Li et al., 2024). Average precision for ranks 5 and 20 for top 20 and 50 cell types is used. AUROC at top 1, 10, 20, and 50 cell types is used.

Model	AP@5 Top-20 CT	AP@20 Top-20 CT	AP@5 Top-50 CT	AP@20 Top-50 CT
PINNACLE (RA)	0.917±0.057	0.625 ±0.041	0.628 ±0.057	0.452 ±0.033
GATv2 (RA)	0.334±0.050	0.246±0.055	0.252±0.021	0.187±0.033
Node2Vec (RA)	0.585±0.045	0.448 ±0.023	0.476 ±0.028	0.383 ±0.020
Label Propagation (RA)	0.105±0.024	0.077±0.005	0.042±0.010	0.048±0.005
PINNACLE (IBD)	0.884±0.071	0.634 ±0.048	0.582 ±0.047	0.439 ±0.030
GATv2 (IBD)	0.348±0.043	0.261±0.007	0.259±0.017	0.189±0.004
Node2Vec (IBD)	0.617±0.081	0.417±0.044	0.406 ±0.070	0.295±0.036
Label Propagation (IBD)	0.268±0.014	0.228±0.005	0.126±0.011	0.134±0.003
Model	AUROC Top-1 CT	AUROC Top-10 CT	AUROC Top-20 CT	AUROC Top-50 CT
PINNACLE (RA)	0.765±0.054	0.676±0.017	0.652±0.013	0.604 ±0.007
GATv2 (RA)	0.5	0.5	0.5	0.5
Node2Vec (RA)	0.766 ±0.030	0.731 ±0.024	0.715 ±0.025	0.687 ±0.024
Label Propagation (RA)	0.451±0.003	0.415±0.002	0.392±0.002	0.364±0.002
PINNACLE (IBD)	0.935±0.067	0.806±0.023	0.757±0.017	0.672 ±0.015
GATv2 (IBD)	0.5	0.5	0.5	0.5
Node2Vec (IBD)	0.883 ±0.116	0.703 ±0.036	0.666 ±0.030	0.610±0.031
Label Propagation (IBD)	0.683±0.000	0.544±0.000	0.511±0.001	0.457±0.000

Our results (table 2) show Brody et al. (2022); Grover & Leskovec (2016) extract non-random, informative signals by their outperforming label propagation at retrieval, but both methods are far from optimal. SoTA general ML models and domain-specific methods performed poorly at this study's single-cell drug target nomination task, supporting the need for stronger models. Li et al. (2024), which outperformed all methods, is a first step in this direction. Still, it is specifically engineered for cell type-specific PPI networks and can only work with apriori-defined and user-specific contexts. In addition, Li et al. (2024)'s contextualized AUROC was strongest at IBD, where label propagation also performed well (better than (Brody et al., 2022)), but weaker and even inferior to Grover & Leskovec (2016) in RA, where label propagation performed worse than random. This suggests ample room for improvement at generalization out of subgraph-based view Mao et al. (2024), and a promising avenue of research in this task for GFMs and other representation-learning-based transfer learning methods.

5. Conclusion

PyTDC represents a significant step in accelearting scientific research by developing machine learning infrastructure (Kahn et al., 2022) for biomedical AI, particularly in single-cell therapeutics (Theodoris, 2024). By integrating multimodal data retrieval (section 3.1), standardized benchmarking (sections 4 and A), and model inference (sections 3.2 and C) into a unified platform, PyTDC enables seamless access to heterogeneous biological datasets and provides a foundation for training and evaluating machine learning models in therapeutics. Introducing an API-first architecture (Beaulieu et al., 2022) facilitates continuous data updates, ensuring that biomedical AI models remain adaptable to discoveries. Moreover, PyTDC's model server (figures 2 and 1) streamlines biomedical foundation models' retrieval, deployment, and fine-tuning, aligning new method development with real-world therapeutic objectives. Our case study on single-cell drug-target nomination highlights key challenges in model generalization and domain-specific evaluation. Despite advances in graph representation learning and context-aware geometric deep learning, existing methods struggle to generalize across unseen cell types and integrate additional biomedical modalities. The introduction of context-specific evaluation metrics within PyTDC provides a quantitative framework for assessing model performance in the most predictive biological contexts, addressing critical gaps in therapeutic model benchmarking. These results emphasize the need for multimodal, context-aware foundation models, a research direction that PyTDC is uniquely positioned to support. By defining open challenges, establishing state-of-the-art baselines, and providing tools for method development and evaluation, PyTDC fosters a more rigorous, reproducible, and therapeutically relevant machine learning ecosystem. PyTDC will accelerate innovation in single-cell AI, drive improvements in domain-specific generalization, and establish new standards for biomedical machine learning research. We invite the community to contribute to this open-source effort and leverage PyTDC as a scalable, adaptable, and transparent framework for advancing therapeutics through AI.

Software and Data

PyTDC is open-source at https://github.com/apliko-xyz/PyTDC and documentation is provided via https://pytdc.apliko.io. In addition, notebooks used to perform the benchmarking discussed in this paper can be found at https://github.com/amva13/pinnacle_benchmark.

Acknowledgements

We thank Jesus Caraballo Anaya, a member of the PyTDC team, for his contributions to the technologies discussed in this manuscript. We thank the TDC team for collecting useful datasets.

We gratefully acknowledge the support of NIH R01-HD108794, NSF CAREER 2339524, US DoD FA8702-15-D-0001, awards from Harvard Data Science Initiative, Amazon Faculty Research, Google Research Scholar Program, AstraZeneca Research, Roche Alliance with Distinguished Scientists, Sanofi iDEA-iTECH Award, Pfizer Research, Chan Zuckerberg Initiative, John and Virginia Kaneb Fellowship award at Harvard Medical School, Aligning Science Across Parkinson's (ASAP) Initiative, Biswas Computational Biology Initiative in partnership with the Milken Institute, Harvard Medical School Dean's Innovation Awards for the Use of Artificial Intelligence, and Kempner Institute for the Study of Natural and Artificial Intelligence at Harvard University. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funders.

Impact Statement

Limitations and societal considerations. PyTDC software is open-source and available under MIT license. However, models and datasets exposed by PyTDC are released under their own licenses, which though exposed, PyTDC is unable to strictly enforce. It's been the case such licenses have been violated by commercial entities developing models on released datasets (Zambrano Chaves et al., 2024). Open-source datasets and benchmark can increase risk of biased or incomplete data, which may lead to inaccurate or non-representative AI models. Additionally, the open accessibility of such datasets could lead to misuse, including unethical applications or the proliferation of AI models that reinforce existing biases in medicine. Moreover, reliance on standardized benchmarks may discourage innovation and lead to the over-fitting of models to specific datasets, potentially limiting their generalization in real-world scenarios. Last, evaluating deep learning models for genetic perturbation tasks requires reconsideration in light of recent findings that question their effectiveness for this problem. A recent study revealed that deep learning models do not consistently outperform simpler linear models across various benchmarks (Ahlmann-Eltze et al., 2024). PyTDC datasets, benchmarks, metrics, and foundation model tooling lay the foundation for more thorough study, development, and evaluation of models in this space.

References

Agrawal, M., Zitnik, M., and Leskovec, J. Large-scale analysis of disease pathways in the human interactome. In *Pacific Symposium on Biocomputing 2018*, pp. 111–122. World Scientific, 2018.

Agrawal, P., Gopalan, V., and Hannenhalli, S. Predicting gene expression changes upon epigenomic drug treatment. *bioRxiv*, 2023. doi: 10.1101/2023.07.20.549955. URL https://doi.org/10.1101/2023.07.20.549955.

Ahlmann-Eltze, C., Huber, W., and Anders, S. Deep learning-based predictions of gene perturbation effects do not yet outperform simple linear methods. *bioRxiv*, 2024. doi: 10.1101/2024.09.16.613342. URL https://www.biorxiv.org/content/early/2024/10/28/2024.09.16.613342.

Aissa, A. F., Islam, A. B. M. M. K., Ariss, M. M., Go, C. C., Rader, A. E., Conrardy, R. D., Gajda, A. M., Rubio-Perez, C., Valyi-Nagy, K., Pasquinelli, M., Feldman, L. E., Green, S. J., Lopez-Bigas, N., Frolov, M. V., and Benevolenskaya, E. V. Single-cell transcriptional changes associated with drug tolerance and response to combination therapies in cancer. *Nature Communications*, 12(1): 1628, 2021. doi: 10.1038/s41467-021-21761-6. URL https://doi.org/10.1038/s41467-021-21761-6.

Aridhi, S., Alborzi, S. Z., Smaïl-Tabbone, M., Devignes, M.-D., and Ritchie, D. Neighborhood-based label propagation in large protein graphs. *arXiv preprint arXiv:1708.07074*, 2017. doi: 10.48550/arXiv.1708.07074.

Arvind, V., Kim, J. S., Oermann, E., Kaji, D. A., and Cho, S. K. Predicting surgical complications in adult patients undergoing anterior cervical discectomy and fusion using machine learning. *Neurospine*, 15:329–337, 2018. doi: 10.14245/ns.1836248.124.

Asadi, H., Dowling, R., Yan, B., and Mitchell, P. Machine learning for outcome prediction of acute ischemic stroke post intra-arterial therapy. *PLoS ONE*, 9:e88225, 2014. doi: 10.1371/journal.pone.0088225.

Beacher, F., Mujica-Parodi, L., Gupta, S., and Ancora, L. A. Machine learning predicts outcomes of phase iii clinical trials for prostate cancer. *Algorithms*, 14:147, 2021. doi: 10.3390/a14050147.

Beaulieu, N., Dascalu, S., and Hand, E. Api integrator: A ui design and code automation application supporting api-first design. In *Proceedings of the 9th International Conference on Applied Computing and Information Technology*, 2022. doi: 10.1145/3543895.3543939.

Bedon, L., Cecchin, E., Fabbiani, E., Bo, M. D., Buonadonna, A., Polano, M., and Toffoli, G. Machine learning application in a phase i clinical trial allows for the identification of clinical-biomolecular markers significantly associated with toxicity. *Clinical Pharmacology and Therapeutics*, 111, 2021. doi: 10.1002/cpt.2511.

Beinse, G., Tellier, V., Charvet, V., Deutsch, E., Borget, I., Massard, C., Hollebecque, A., and Verlingue, L. Prediction of drug approval after phase i clinical trials in oncology: Resolved2. *JCO Clinical Cancer Informatics*, 3:1–10, 2019. doi: 10.1200/CCI.19.00023.

Bhasin, M., Lata, S., and Raghava, G. P. S. Tappred prediction of tap-binding peptides in antigens. *Methods in Molecular Biology*, 409:381–386, 2007. doi: 10.1007/978-1-60327-118-9_28.

Boulanger, D., Eccleston, R. C., Phillips, A., Coveney, P., Elliott, T., and Dalchau, N. A mechanistic model for predicting cell surface presentation of competing peptides by mhc class i molecules. *Frontiers in Immunology*, 9: 1538, 2018. doi: 10.3389/fimmu.2018.01538.

Bowman, E., Banuprakash, S., Nguyen, K.-S., and Marini, M. Machine learning prediction of progression events in oncology recist 1.1 clinical trials. *Journal of Clinical Oncology*, 2023. doi: 10.1200/jco.2023.41.16_suppl.e13589.

Bradley, P. Structure-based prediction of t cell receptor:peptide-mhc interactions. *eLife*, 12, 2022. doi: 10.1101/2022.08.05.503004.

Brbi, M., Yasunaga, M., Agarwal, P., and Leskovec, J. Predicting drug outcome of population via clinical knowledge graph. *To be published*, 2024. URL https://www.medrxiv.org/content/10.1101/2024.03.06.24303800v1.full.pdf. Preprint.

Brody, S., Alon, U., and Yahav, E. How attentive are graph attention networks? In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=F72ximsx7C.

Brown, J. S., Mohapatra, S., Lee, M. A., Misteli, R., Tseo, Y., Grob, N. M., Quartararo, A. J., Loas, A., Gomez-Bombarelli, R., and Pentelute, B. L. Unsupervised machine learning leads to an abiotic picomolar peptide ligand. *ChemRxiv*, May 2023. doi: 10.26434/chemrxiv-2023-tws4n. URL https://www.researchgate.net/publication/371033601_Unsupervised_machine_learning_leads_to_an_abiotic_picomolar_peptide_ligand. License CC BY-NC-ND 4.0.

Bucanek, J. *Learn Objective-C for Java Developers*, chapter 20, pp. 353. Apress, 01 2009. ISBN 978-1-4302-2369-6. doi: 10.1007/978-1-4302-2370-2_20.

Cai, M., Bang, S.-J., Zhang, P., and Lee, H. Atm-tcr: Tcr-epitope binding affinity prediction using a multi-head self-attention model. *Frontiers in Immunology*, 13, 2022. doi: 10.3389/fimmu.2022.893247.

Carlberg, K., Korotkova, M., Larsson, L., Catrina, A., Ståhl, P. L., and Malmström, V. Exploring inflammatory signatures in arthritic joint biopsies with spatial transcriptomics. *Scientific Reports*, 9, 2019. doi: 10.1038/s41598-019-55441-y.

Chandak, P., Huang, K., and Zitnik, M. Building a knowledge graph to enable precision medicine. *Scientific Data*, 10(1):67, 2023. URL https://doi.org/10.1038/s41597-023-01960-3.

Chen, V. S., Wu, S., Weng, Z., Ratner, A., and Ré, C. Slice-based learning: A programming model for residual learning in critical data slices, 2020a. URL https://arxiv.org/abs/1909.06349.

Chen, X., Wang, Z., Zitnik, M., Kellis, M., and Chen, T. Pre-training of single-cell language models through genetic pathway learning. In *Workshop on Accessible and Efficient Foundation Models for Biological Discovery, ICML*, June 2024. URL https://icml.cc/virtual/2024/35927. AccMLBio Poster.

Chen, Z., Min, M. R., and Ning, X. Ranking-based convolutional neural network models for peptide-mhc binding prediction. *ArXiv*, abs/2012.02840, 2020b.

Churi, P. P., Wagh, S., Kalelkar, D., and Kalelkar, M. Model-view-controller pattern in bi dashboards: Designing best practices. *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 2082–2086, 2016.

Cock, P. J., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B., and de Hoon, M. J. L. Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423, 2009. URL https://biopython.org/docs/1.75/api/Bio.Entrez.htmlandhttps://biopython.org/docs/1.75/api/Bio.SeqIO.html.

Consortium, T. U. Uniprot: the universal protein knowledgebase in 2021. *Nucleic Acids Research*, 49(D1): D480–D489, 2021. doi: 10.1093/nar/gkaa1100. URL https://www.uniprot.org/help/fasta-headers.

Cui, H., Wang, C., Maan, H., Pang, K., Luo, F., Duan, N., and Wang, B. scgpt: toward building a foundation model for single-cell multi-omics using generative ai. *Nature Methods*, pp. 1–11, 2024.

CZI Single-Cell Biology, et al. Cz cellxgene discover: A single-cell data platform for scalable exploration, analysis and modeling of aggregated data. bioRxiv Preprint, 2023. URL https://doi.org/10.1101/2023.10.30.563174.

Dai, R., Kannampallil, T., Zhang, J., Lv, N., Ma, J., and Lu, C. Multi-task learning for randomized controlled trials. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6:1–23, 2022. doi: 10.1145/3534591.

Ektefaie, Y., Shen, A., Bykova, D., Maximillian, M., Zitnik, M., and Farhat, M. Evaluating generalizability of artificial intelligence models for molecular datasets. *Nature Machine Intelligence*, 2024. URL https://rdcu.be/d2D0z.

Emmerik, M. V., Rappoport, A., and Rossignac, J. Simplifying interactive design of solid models: A hypertext approach. *The Visual Computer*, 9:239–254, 1993. doi: 10.1007/BF01908447.

ESM Team. ESM Cambrian: Revealing the mysteries of proteins with unsupervised learning. EvolutionaryScale Website, December 2024. URL https://evolutionaryscale.ai/blog/esm-cambrian. Accessed: January 29, 2024.

Fu, T., Huang, K., Xiao, C., Glass, L. M., and Sun, J. Hint: Hierarchical interaction network for clinical-trial-outcome predictions. *Patterns*, 3(4):100445, Feb 2022. doi: 10.1016/j.patter.2022.100445. URL https://doi.org/10.1016/j.patter.2022.100445. eCollection 2022 Apr 8.

Fujima, N., Shimizu, Y., Yoshida, D., Kano, S., Mizumachi, T., Homma, A., Yasuda, K., Onimaru, R., Sakai, O., Kudo, K., and Shirato, H. Machine learning-based prediction of treatment outcomes using mr imaging-derived quantitative tumor information in patients with sinonasal squamous cell carcinomas: A preliminary study. *Cancers*, 11: 800, 2019. doi: 10.3390/cancers11060800.

Gao, J., Xiao, C., Glass, L. M., and Sun, J. Compose: Cross-modal pseudo-siamese network for patient trial matching. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 803–812, 2020.

Gao, S., Fang, A., Huang, Y., Giunchiglia, V., Noori, A., Schwarz, J. R., Ektefaie, Y., Kondic, J., and Zitnik, M. Empowering biomedical discovery with ai agents. *CellPress*, 187, 2024.

Gao, Y., Gao, Y., and Liu, Q. Pan-peptide meta learning for t-cell receptor–antigen binding recognition. *Nature Machine Intelligence*, 5:236–249, 2023. doi: 10.1038/s42256-023-00619-3. URL https://www.nature.com/articles/s42256-023-00619-3.

Garau-Luis, J. J., Bordes, P., Gonzalez, L., Roller, M., de Almeida, B. P., Hexemer, L., Blum, C., Laurent, S., Grzegorzewski, J., Lang, M., Pierrot, T., and Richard, G. Multi-modal transfer learning between biological foundation models, 2024. URL https://arxiv.org/abs/2406.14150.

Ghadie, M. A., Lambourne, L., Vidal, M., and Xia, Y. Domain-based prediction of the human isoform interactome provides insights into the functional impact of alternative splicing. *PLoS Computational Biology*, 13, 2017. doi: 10.1371/journal.pcbi.1005717.

Grazioli, F., Machart, P., Mösch, A., Li, K., Castorina, L., Pfeifer, N., and Min, M. R. Attentive variational information bottleneck for tcr–peptide interaction prediction. *Bioinformatics*, 39, 2022a. doi: 10.1093/bioinformatics/btac820.

Grazioli, F., Mösch, A., Machart, P., Li, K., Alqassem, I., O'Donnell, T. J., and Min, M. R. On tcr binding predictors failing to generalize to unseen peptides. *Frontiers in Immunology*, 13:1014256, 2022b. doi: 10.3389/fimmu.2022.1014256. URL https://www.frontiersin.org/articles/10.3389/fimmu.2022.1014256/full.

Grover, A. and Leskovec, J. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 855–864. ACM, 2016. doi: 10.1145/2939672.2939754.

Gupta, C., Xu, J., Jin, T., Khullar, S., Liu, X., Alatkar, S., Cheng, F., and Wang, D. Single-cell network biology characterizes cell type gene regulation for drug repurposing and phenotype prediction in alzheimer's disease. *PLoS Computational Biology*, 18, 2022. doi: 10.1371/journal.pcbi.1010287.

Hao, M., Gong, J., Zeng, X., Liu, C., Guo, Y., Cheng, X., Wang, T., Ma, J., Zhang, X., and Song, L. Large-scale foundation model on single-cell transcriptomics. *Nature Methods*, pp. 1–11, 2024.

Hetzel, L., Böhm, S., Kilbertus, N., Günnemann, S., Lotfollahi, M., and Theis, F. Predicting cellular responses to novel drug perturbations at a single-cell resolution. *arXiv*, abs/2204.13545, 2022. URL https://arxiv.org/abs/2204.13545.

Hong, D., Fort, D., Shi, L., and Price-Haywood, E. Electronic medical record risk modeling of cardiovascular outcomes among patients with type 2 diabetes. *Diabetes Therapy*, 12:2007 – 2017, 2021. doi: 10.1007/s13300-021-01096-w.

Hoofnagle, A. N. and Resing, K. A. Proteomics and the analysis of protein phosphorylation. *Current Opinion in Biotechnology*, 12(6):617–622, 2001.

Hu, X. and Liu, S. Deepbcr: Deep learning framework for cancer-type classification and binding affinity estimation using b cell receptor repertoires. *bioRxiv*, 2019. doi: 10.1101/731158.

Huang, K., Fu, T., Gao, W., Zhao, Y., Roohani, Y., Leskovec, J., Coley, C. W., Xiao, C., Sun, J., and Zitnik, M. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development, 2021.

Huang, K., Fu, T., Gao, W., Zhao, Y., Roohani, Y., Leskovec, J., Coley, C. W., Xiao, C., Sun, J., and Zitnik, M. Artificial intelligence foundation for therapeutic science. *Nature chemical biology*, 18(10):1033–1036, 2022.

Huang, Q., Ren, H., Chen, P., Kržmanc, G., Zeng, D., Liang, P., and Leskovec, J. PRODIGY: Enabling In-context Learning Over Graphs. In *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS)*, 2023. URL https://arxiv.org/abs/2310.01408.

Huang, S. and Ding, Y. Predicting binding affinity between mhc-i receptor and peptides based on molecular docking and protein-peptide interaction interface characteristics. *Letters in Drug Design and Discovery*, 2022. doi: 10.2174/1570180819666220819102035.

Initiative, C. Z. Embedding metrics in the december 2023 lts, 2023. URL https://chanzuckerberg.github.io/cellxgene-census/articles/2024/20240710_embedding_metrics_dec_2023_lts.html#accessing-the-embeddings-included-in-the-benchmark. Accessed: 2024-09-14.

Janeway, C. A., Travers, P., Walport, M., and Shlomchik, M. J. *Immunobiology: The Immune System in Health and Disease*. Garland Science, 2001.

Jensen, M. F. and Nielsen, M. Nettcr 2.2 - improved tcr specificity predictions by combining pan- and peptide-specific training strategies, loss-scaling and integration of sequence similarity. *bioRxiv*, 2023. doi: 10.1101/2023.10.12.562001.

Ji, Y., Mishra, R. K., and Davuluri, R. In silico analysis of alternative splicing on drug-target gene interactions. *Scientific Reports*, 10, 2020. doi: 10.1038/s41598-019-56894-x.

Jiang, Y., Huo, M., and Li, S. C. Teinet: a deep learning framework for prediction of tcr-epitope binding specificity. *Briefings in bioinformatics*, 2023. doi: 10.1093/bib/bbad086.

Jones, R. C., Karkanias, J., Krasnow, M., Pisco, A., Quake, S., Salzman, J., Yosef, N., Bulthaup, B., Brown, P., Harper, W., Hemenez, M., Ponnusamy, R., Salehi, A., Sanagavarapu, B. A., Spallino, E., Aaron, K. A., Concepcion, W., Gardner, J., Kelly, B., Neidlinger, N., Wang, Z., Crasta, S., Kolluru, S., Morri, M., Tan, S. Y., Travaglini, K., Xu, C. A., Alcántara-Hernández, M., Almanzar, N., Antony, J., Beyersdorf, B., Burhan, D., Calcuttawala, K., Carter, M. M., Chan, C. K. F., Chang, C. A., Chang, S., Colville, A., Culver, R., Cvijovic, I., D'Amato, G., Ezran, C., Galdos, F. X., Gillich, A., Goodyer, W., Hang, Y., Hayashi, A., Houshdaran, S., Huang, X., Irwin, J., Jang, S., Juanico, J. V., Kershner, A. M., Kim, S., Kiss, B., Kong, W., Kumar, M. E., Kuo, A., Leylek, R., Li, B., Loeb, G. B., Lu, W.-J., Mantri, S., Markovic, M., McAlpine, P. L., de Morrée, A., Mrouj, K., Mukherjee, S., Muser, T., Neuhöfer, P., Nguyen, T. D., Perez, K., Phansalkar, R., Puluca, N., Qi, Z., Rao, P., Raquer-McKay, H. M., Schaum, N., Scott, B., Seddighzadeh, B., Segal, J., Sen, S., Sikandar, S. S., Spencer, S., Steffes, L., Subramaniam, V., Swarup, A., Swift, M., Van Treuren, W. W., Trimm, E., Veizades, S., Vijayakumar, S., Vo, K. C., Vorperian, S., Wang, W., Weinstein, H. N., Winkler, J., Wu, T., Xie, J., Yung, A., Zhang, Y., Detweiler, A., Mekonen, H. E., Neff, N., Sit, R., Tan, M., Yan, J.-c., Bean, G., Charu, V., Forgó, E., Martin, B. A., Ozawa, M., Silva, O., Toland, A., Vemuri, V. N. P., Afik, S., Awayan, K., Botvinnik, O., Byrne, A., Chen, M., Dehghannasiri, R., Gayoso, A., Granados, A. A., Li, Q., Mahmoudabadi, G., McGeever, A., Olivieri, J., Park, M., Ravikumar, N., Stanley, G. M., Tan, W., Tarashansky, A. J., Vanheusden, R., Wang, P. L., Wang, S., Xing, G., Culver, R., Dethlefsen, L., Ho, P.-y., Liu, S., Maltzman, J., Metzger, R., Sasagawa, K., Sinha, R., Song, H., Wang, B., Artandi, S., Beachy, P., Clarke, M., Giudice, L., Huang, F., Huang, K. C., Idoyaga, J., Kim, S. K., Krasnow, M., Kuo, C., Nguyen, P., Rando, T., Red-Horse, K., Reiter, J., Relman, D., Sonnenburg, J., Wu, A., Wu, S. M., and Wyss-Coray, T. The tabula sapiens: A multiple-organ, single-cell transcriptomic atlas of humans. *Science*, 376, 2022. doi: 10.1126/science.abl4896.

Jourahmad, Z., Habibabadi, J. M., Moein, H., Basiratnia, R., Geranqayeh, A. R., Ghidary, S. S., and Sadegh-Zadeh, S.-A. Machine learning techniques for predicting the short-term outcome of resective surgery in lesional-drug resistance epilepsy. *ArXiv*, abs/2302.10901, 2023.

Julkunen, H., Cichońska, A., Gautam, P., Szedmák, S., Douat, J., Pahikkala, T., Aittokallio, T., and Rousu, J. Leveraging multi-way interactions for systematic prediction of pre-clinical drug combination effects. *Nature Communications*, 11, 2020. doi: 10.1038/s41467-020-19950-z.

Kahn, J. D., Pratap, V., Likhomanenko, T., Xu, Q., Hannun, A., Cai, J., Tomasello, P., Lee, A., Grave, E., Avidov, G., Steiner, B., Liptchinsky, V., Synnaeve, G., and Collobert, R. Flashlight: Enabling innovation in tools for machine learning. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 10557–10574. PMLR, 2022. URL https://proceedings.mlr.press/v162/kahn22a.html.

Kalscheur, M. M., Kipp, R., Tattersall, M., Mei, C., Buhr, K., DeMets, D., Field, M., Eckhardt, L., and Page, C. D. Machine learning algorithm predicts cardiac resynchronization therapy outcomes: Lessons from the companion trial. *Circulation: Arrhythmia and Electrophysiology*, 11: e005499, 2018. doi: 10.1161/CIRCEP.117.005499.

Kim, H. Y., Kim, S., Park, W.-Y., and Kim, D. Tspred: a robust prediction framework for tcr-epitope interactions based on an ensemble deep learning approach using paired chain tcr sequence data. *bioRxiv*, 2023. doi: 10.1101/2023.12.04.570002.

Klein, D. sciplex dataset. Dataset, 2023. URL https://doi.org/10.6084/m9.figshare.24681285.v1.

Koşaloğlu-Yalçın, Z., Lee, J., Nielsen, M., Greenbaum, J., Schoenberger, S., Miller, A. M., Kim, Y. J., Sette, A., and Peters, B. Combined assessment of mhc binding and antigen expression improves t cell epitope predictions. *bioRxiv*, 2020. doi: 10.1101/2020.11.09.375204.

Kuenzi, B., Park, J., Fong, S. H., Sanchez, K. S., Lee, J., Kreisberg, J., Ma, J., and Ideker, T. Predicting drug response and synergy using a deep learning model of human cancer cells. *Cancer cell*, 2020. doi: 10.1016/j.ccell.2020.09.014.

Kwon, J. J., Pan, J., Gonzalez, G., Hahn, W. C., and Zitnik, M. On knowing a gene: A distributional hypothesis of gene function. *Cell Systems*, 2024. doi: 10.1016/j.cels.2024.04.008.

Lee, I., Keum, J., and Nam, H. Deepconv-dti: Prediction of drug-target interactions via deep learning with convolution on protein sequences. *PLoS Computational Biology*, 15, 2018. doi: 10.1371/journal.pcbi.1007129.

Li, M. M., Huang, Y., Sumathipala, M., Liang, M. Q., Valdeolivas, A., Ananthakrishnan, A. N., Marbach, D., and Zitnik, M. Contextual ai models for single-cell protein biology. *Nature Methods*, 2024. URL https://www.nature.com/articles/s41592-024-02341-3.

Li, S., Zhou, J., Xu, T., Huang, L., Wang, F., Xiong, H., Huang, W., Dou, D., and Xiong, H. Structure-aware interactive graph neural networks for the prediction of protein-ligand binding affinity. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2021. doi: 10.1145/3447548.3467311.

Li, X., Wang, S., Zhang, J., and Zhang, R. Torchprotein: A deep learning library for protein sequence and structure modeling. *Bioinformatics*, 38(6):1743–1745, 2022. doi: 10.1093/bioinformatics/btab848.

Li, Y., Liu, X., You, Z., Li, L., Guo, J., and Wang, Z. A computational approach for predicting drug–target interactions from protein sequence and drug substructure fingerprint information. *International Journal of Intelligent Systems*, 36:593 – 609, 2020. doi: 10.1002/int.22332.

Li, Y.-M., Huang, Y.-A., You, Z., Li, L., and Wang, Z. Drug-target interaction prediction based on drug fingerprint information and protein sequence. *Molecules*, 24, 2019. doi: 10.3390/molecules24162999.

Liu, H., P.V.S, A., Patwardhan, S., Grasch, P., and Agarwal, S. Model stability with continuous data updates, 2022. URL https://arxiv.org/pdf/2201.05692.pdf.

Liu, N. and Salinas, J. Machine learning for predicting outcomes in trauma. *SHOCK*, 48:504–510, 2017. doi: 10.1097/SHK.0000000000000898.

Liu, T., Li, K., Wang, Y., Li, H., and Zhao, H. Evaluating the utilities of foundation models in single-cell data analysis. *bioRxiv*, 2023. doi: 10.1101/2023.09.08.555192. URL https://doi.org/10.1101/2023.09.08.555192. Preprint.

Liu, Z., Li, Y., Han, L., Li, J., Liu, J., Zhao, Z., Nie, W., Liu, Y., and Wang, R. Pdb-wide collection of binding data: current status of the pdbbind database. *Bioinformatics*, 31 (3):405–412, 2015. doi: 10.1093/bioinformatics/btu626. URL https://doi.org/10.1093/bioinformatics/btu626.

Liu, Z., Jin, J., Cui, Y., Xiong, Z., Nasiri, A., Zhao, Y., and Hu, J. Deepseqpanii: an interpretable recurrent neural network model with attention mechanism for peptide-hla class ii binding prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2021. doi: 10.1109/TCBB.2021.3074927.

Lopez, R., Regier, J., Cole, M. B., Jordan, M. I., and Yosef, N. Deep generative modeling for single-cell transcriptomics. *Nature Methods*, 15(12):1053–1058,

2018. doi: 10.1038/s41592-018-0229-2. URL https://www.nature.com/articles/s41592-018-0229-2.

Lotfollahi, M., Wolf, F., and Theis, F. J. scgen predicts single-cell perturbation responses. *Nature Methods*, 16: 715–721, 2019. doi: 10.1038.

Luecken, M. D., Gigante, S., Burkhardt, D. B., Cannoodt, R., Strobl, D. C., Markov, N. S., Zappia, L., Palla, G., Lewis, W., Dimitrov, D., Vinyard, M. E., Magruder, D., Andersson, A., Dann, E., Qin, Q., Otto, D. J., Klein, M., Botvinnik, O. B., Deconinck, L., Waldrant, K., Members, O. P. J., Bloom, J. M., Pisco, A. O., Saez-Rodriguez, J., Wulsin, D., Pinello, L., Saeys, Y., Theis, F. J., and Krishnaswamy, S. Defining and benchmarking open problems in single-cell analysis. *Research Square Preprint*, 2023. doi: 10.21203/rs.3.rs-4181617/v1.

Lupia, A., Mimmi, S., Iaccino, E., Maisano, D., Moraca, F., Talarico, C., Vecchio, E., Fiume, G., Ortuso, F., Scala, G., Quinto, I., and Alcaro, S. Molecular modelling of epitopes recognized by neoplastic b lymphocytes in chronic lymphocytic leukemia. *European Journal of Medicinal Chemistry*, 111838, 2019. doi: 10.1016/j.ejmech.2019.111838.

Lv, H., Yang, X., Wang, B., Wang, S., Du, X., Tan, Q., Hao, Z., Liu, Y., Yan, J., and Xia, Y. Machine learning–driven models to predict prognostic outcomes in patients hospitalized with heart failure using electronic health records: Retrospective study. *Journal of Medical Internet Research*, 23, 2020. doi: 10.2196/24996.

M. Bran, A., Cox, S., Schilter, O., Baldassari, C., White, A. D., and Schwaller, P. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, 2024.

MacKay, E. J., Stubna, M. D., Chivers, C., Draugelis, M., Hanson, W. J., Desai, N. D., and Groeneveld, P. W. Application of machine learning approaches to administrative claims data to predict clinical outcomes in medical and surgical patient populations. *PLoS ONE*, 16, 2021. doi: 10.1371/journal.pone.0252585.

Malik, V. K., Pathak, S., Anamika, K., Kaur, A., and Kumar, V. A study of mvc: A software design pattern for web application development on j2ee architecture. *Academia.edu*, 2021. URL https://www.academia.edu/50941063/A_STUDY_OF_MVC_A_SOFTWARE_DESIGN_PATTERN_FOR_WEB_APPLICATION_DEVELOPMENT_ON_J2EE_ARCHITECTURE.

Mao, H., Chen, Z., Tang, W., Zhao, J., Ma, Y., Zhao, T., Shah, N., Galkin, M., and Tang, J. Position: Graph foundation models are already here. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024. URL https://arxiv.org/abs/2402.02216.

Membarth, R., Reiche, O., Hannig, F., Teich, J., Körner, M., and Eckert, W. Hipacc: A domain-specific language and compiler for image processing. *IEEE Transactions on Parallel and Distributed Systems*, 27:210–224, 2016. doi: 10.1109/TPDS.2015.2394802.

Meng, F.-R., You, Z.-H., Chen, X., Zhou, Y., and An, J.-Y. Prediction of drug–target interaction networks from the integration of protein sequences and drug chemical structures. *Molecules*, 22(7), 2017. ISSN 1420-3049. doi: 10.3390/molecules22071119. URL https://www.mdpi.com/1420-3049/22/7/1119.

Meslamani, J., Rognan, D., and Kellenberger, E. sc-pdb: a database for identifying variations and multiplicity of 'druggable' binding sites in proteins. *Bioinformatics*, 27 (9):1324–1326, 2011.

Moris, P., Pauw, J. D., Postovskaya, A., Gielis, S., Neuter, N. D., Bittremieux, W., Ogunjimi, B., Laukens, K., and Meysman, P. Current challenges for unseen-epitope tcr interaction prediction and a new perspective derived from image classification. *Briefings in Bioinformatics*, 22, 2020. doi: 10.1093/bib/bbaa318.

Murphy, K. and Weaver, C. *Janeway's Immunobiology*. Garland Science, 2016.

Mysinger, M. M., Carchia, M., Irwin, J. J., and Shoichet, B. K. Directory of useful decoys, enhanced (dud-e): better ligands and decoys for better benchmarking. *Journal of medicinal chemistry*, 55(14):6582–6594, 2012.

Nair, V., Saleh, R., Toor, S., Taha, R. Z., Ahmed, A., Kurer, M., Murshed, K. A., Nada, M., and Elkord, E. Epigenetic regulation of immune checkpoints and t cell exhaustion markers in tumor-infiltrating t cells of colorectal cancer patients. *Epigenomics*, 12(17):1481–1492, 2020. doi: 10.2217/epi-2020-0267. URL https://doi.org/10.2217/epi-2020-0267.

Nielsen, M., Lundegaard, C., Lund, O., and Keşmir, C. The role of the proteasome in generating cytotoxic t-cell epitopes: insights obtained from improved predictions of proteasomal cleavage. *Immunogenetics*, 57:33–41, 2005. doi: 10.1007/s00251-005-0781-7.

Norman, T. M., Horlbeck, M. A., Replogle, J. M., Ge, A. Y., Xu, A., Jost, M., Gilbert, L. A., and Weissman, J. S. Exploring genetic interaction manifolds constructed from rich single-cell phenotypes. *Science*, 365(6455):786–793, 2019. doi: 10.1126/science.aax4438.

Öztürk, H., Ozkirimli, E., and Özgür, A. Widedta: prediction of drug-target binding affinity. *arXiv preprint arXiv:1902.04166*, 2019. URL https://doi.org/10.48550/arXiv.1902.04166.

O'Donnell, T., Rubinsteyn, A., and Laserson, U. Mhcflurry 2.0: Improved pan-allele prediction of mhc class i-presented peptides by incorporating antigen processing. *Cell Systems*, 11(1):42–48.e7, 2020. doi: 10.1016/j.cels.2020.06.010.

Papadopoulos, S., Datta, K., Madden, S., and Mattson, T. The tiledb array data storage manager. *Proc. VLDB Endow.*, 10:349–360, 2016. doi: 10.14778/3025111.3025117.

Parca, L., Pepe, G., Pietrosanto, M., Galvan, G., Galli, L., Palmeri, A., Sciandrone, M., Ferrè, F., Ausiello, G., and Helmer-Citterich, M. Modeling cancer drug response through drug-specific informative genes. *Scientific Reports*, 9, 2019. doi: 10.1038/s41598-019-50720-0.

Park, J. H., Cho, H. E., Kim, J. H., Wall, M., Stern, Y., Lim, H., Yoo, S., Kim, H.-S., and Cha, J. Machine learning prediction of incidence of alzheimer's disease using large-scale administrative health data. *NPJ Digital Medicine*, 3, 2020. doi: 10.1038/s41746-020-0256-0.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL https://pytorch.org/docs/stable/data.html#torch.utils.data.IterableDataset.

Patil, S. G., Zhang, T., Wang, X., and Gonzalez, J. E. Gorilla: Large language model connected with massive apis. *ArXiv*, abs/2305.15334, 2023. doi: 10.48550/arXiv.2305.15334.

Peidli, S., Green, T. D., Shen, C., Gross, T., Min, J., Garda, S., Yuan, B., Schumacher, L. J., Taylor-King, J. P., Marks, D. S., Luna, A., Blüthgen, N., and Sander, C. scperturb: harmonized single-cell perturbation data. *Nature Methods*, 21:531–540, Jan 2024.

Piran, Z., Cohen, N., Hoshen, Y., and Nitzan, M. Biological representation disentanglement of single-cell data. *bioRxiv*, 2023. doi: 10.1101/2023.03.05.531195. URL https://doi.org/10.1101/2023.03.05.531195.

Ploegh, H. L. Antigen processing and presentation. *Nature*, 353(6342):125–130, 1998.

Rahimian, F., Salimi-Khorshidi, G., Payberah, A. H., Tran, J., Solares, R. A., Raimondi, F., Nazarzadeh, M., Canoy, D., and Rahimi, K. Predicting the risk of emergency admission with machine learning: Development and validation using linked electronic health records. *PLoS Medicine*, 15, 2018. doi: 10.1371/journal.pmed.1002695.

Rammerstorfer, M. and Mössenböck, H. Data mappings in the model-view-controller pattern. *Broy, M., Zamulin, A.V. (eds) Perspectives of System Informatics. PSI 2003*, pp. 121–132, 2003. doi: 10.1007/978-3-540-39866-0_15.

Replogle, J. M., Saunders, R. A., Pogson, A. N., Jost, M., Norman, T. M., Weissman, J. S., et al. Mapping information-rich genotype-phenotype landscapes with genome-scale perturb-seq. *Cell*, 185(14):2559–2575.e28, 2022. doi: 10.1016/j.cell.2022.05.013. Open access.

Reynisson, B., Alvarez, B., Paul, S., Peters, B., and Nielsen, M. Netmhcpan-4.1 and netmhciipan-4.0: improved predictions of mhc antigen presentation by concurrent motif deconvolution and integration of ms mhc eluted ligand data. *Nucleic Acids Research*, 48:W449 – W454, 2020. doi: 10.1093/nar/gkaa379.

Romero-Molina, S., Ruiz-Blanco, Y. B., Mieres-Perez, J., Harms, M., Münch, J., Ehrmann, M., and Sánchez-García, E. Ppi-affinity: A web tool for the prediction and optimization of protein–peptide and protein–protein binding affinity. *Journal of Proteome Research*, 21:1829 – 1841, 2022. doi: 10.1021/acs.jproteome.2c00020.

Rongali, S., Rose, A., McManus, D., Bajracharya, A. S., Kapoor, A., Granillo, E., and Yu, H. Learning latent space representations to predict patient outcomes: Model development and validation. *Journal of Medical Internet Research*, 22, 2020. doi: 10.2196/16374.

Roohani, Y., Huang, K., and Leskovec, J. Predicting transcriptional outcomes of novel multigene perturbations with gears. *Nature Biotechnology*, Aug 2023. Open access.

Saxena, S., Animesh, S., Fullwood, M., and Mu, Y. Onionmhc: A deep learning model for peptide — hla-a*02:01 binding predictions using both structure and sequence feature sets. *Journal of Micromechanics and Molecular Physics*, 2020. doi: 10.1142/s2424913020500095.

Sayaman, R. W., Wolf, D. M., Yau, C., Wulfkuhle, J., Petricoin, E., Brown-Swigart, L., Asare, S. M., Hirst, G. L., Sit, L., O'Grady, N., Hedistian, D., Consortium, I.-S. . T., Esserman, L. J., LaBarge, M. A., and van 't Veer, L. J. Application of machine learning to elucidate the biology predicting response in the i-spy 2 neoadjuvant breast cancer trial. *Cancer Research*, 80(4 Suppl), 2020. doi: 10.1158/1538-7445.sabcs19-p1-21-08.

Schatz, D. G. and Swanson, P. C. V(d)j recombination: mechanisms of initiation. *Annual Review of Genetics*, 45: 167–202, 2011.

Schick, T., Schmid, H., and Schütze, H. Toolformer: Language models can teach themselves to use tools. In *Proceedings of the 2023 Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2023.

Schperberg, A. V., Boichard, A., Tsigelny, I., Richard, S., and Kurzrock, R. Machine learning model to predict oncologic outcomes for drugs in randomized clinical trials. *International Journal of Cancer*, 147:2537–2549, 2020. doi: 10.1002/ijc.33240.

Senders, J., Staples, P. C., Karhade, A., Zaki, M. M., Gormley, W., Broekman, M., Smith, T., and Arnaout, O. Machine learning and neurosurgical outcome prediction: A systematic review. *World Neurosurgery*, 109:476–486.e1, 2018. doi: 10.1016/j.wneu.2017.09.149.

Shamout, F. E., Zhu, T., and Clifton, D. Machine learning for clinical outcome prediction. *IEEE Reviews in Biomedical Engineering*, 14:116–126, 2020. doi: 10.1109/RBME.2020.3007816.

Siah, K. W., Khozin, S., Wong, C. H., and Lo, A. Machine-learning and stochastic tumor growth models for predicting outcomes in patients with advanced non-small-cell lung cancer. *JCO Clinical Cancer Informatics*, 3:1–11, 2019. doi: 10.1200/CCI.19.00046.

Smith, L. M. and Kelleher, N. L. Proteoform: a single term describing protein complexity. *Nature Methods*, 10(3): 186–187, 2013. doi: 10.1038/nmeth.2369.

Springer, I., Besser, H., Tickotsky-Moskovitz, N., Dvorkin, S., and Louzoun, Y. Prediction of specific tcr-peptide binding from large dictionaries of tcr-peptide pairs. *Frontiers in Immunology*, 11, 2019. doi: 10.3389/fimmu.2020.01803.

Srivatsan, S. R., McFaline-Figueroa, J. L., Ramani, V., Saunders, L., Cao, J., Packer, J., Pliner, H. A., Jackson, D. L., Daza, R. M., Trapnell, C., et al. Massively multiplex chemical transcriptomics at single-cell resolution. *Science*, 367(6473):45–51, 2019. doi: 10.1126/science.aax6234.

T, R., Demerdash, O., and Smith, J. C. Tcr-h: Machine learning prediction of t-cell receptor epitope binding on unseen datasets. *bioRxiv*, 2023. doi: 10.1101/2023.11.28.569077.

Targets, O. Open targets platform: Ra and ibd disease drug targets, 2023. URL https://www.opentargets.org/. Accessed: 2024-05-21.

Tasoluk, B. and Tanrikulu, Z. The performance comparison of a brute-force password cracking algorithm using regular functions and generator functions in python. *International Journal of Security, Privacy and Trust Management*, 2023. doi: 10.5121/ijsptm.2023.12201.

Theodoris, C., Xiao, L., Chopra, A., Chaffin, M., Sayed, Z., Hill, M., Mantineo, H., Brydon, E., Zeng, Z., Liu, S., and Ellinor, P. Transfer learning enables predictions in network biology. *Nature*, 618:1–9, 05 2023. doi: 10.1038/s41586-023-06139-9.

Theodoris, C. V. Perspectives on benchmarking foundation models for network biology. *Quantitative Biology*, 12(4): 335–338, December 2024. doi: 10.1016/j.quantbio.2024.04.001. URL https://doi.org/10.1016/j.quantbio.2024.04.001. Open Access.

Thiago, R., Souza, R., Azevedo, L., Soares, E., Santos, R., Santos, W., Bayser, M. D., Cardoso, M., Moreno, M., and Cerqueira, R. Managing data lineage of o&g machine learning models: The sweet spot for shale use case, 2020. URL https://arxiv.org/abs/2003.04915.

van der Weide, T., Papadopoulos, D., Smirnov, O., Zielinski, M., and van Kasteren, T. Versioning for end-to-end machine learning pipelines. In *Proceedings of the 1st Workshop on Data Management for End-to-End Machine Learning*, DEEM'17, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450350266. doi: 10.1145/3076246.3076248. URL https://doi.org/10.1145/3076246.3076248.

van Os, H., Ramos, L. A., Hilbert, A., van Leeuwen, M., van Walderveen, M. V., Kruyt, N., Dippel, D., Steyerberg, E., van der Schaaf, I., Lingsma, H. F., Schonewille, W., Majoie, C., Olabarriaga, S., Zwinderman, K., Venema, E., Marquering, H., and Wermer, M. Predicting outcome of endovascular treatment for acute ischemic stroke: Potential value of machine learning algorithms. *Frontiers in Neurology*, 9:784, 2018. doi: 10.3389/fneur.2018.00784.

Velez-Arce, A., Lin, X., Huang, K., Li, M. M., Gao, W., Pentelute, B., Fu, T., Kellis, M., and Zitnik, M. Signals in the cells: Multimodal and contextualized machine learning foundations for therapeutics. In *NeurIPS 2024 Workshop on AI for New Drug Modalities*, 2024. URL https://openreview.net/forum?id=kL8dlYp6IM.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio', P., and Bengio, Y. Graph attention networks. *ArXiv*, abs/1710.10903, 2017. doi: 10.17863/CAM.48429.

Wang, J., Zhang, L., Zeng, A., Xia, D., Yu, J., and Yu, G. Deepiii: Predicting isoform-isoform interactions by deep neural networks and data fusion. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19:2177–2187, 2021a. doi: 10.1109/TCBB.2021.3068875.

Wang, Y., Zeng, J., Yang, J., Zhou, J., Niu, B., and Guan, J. Prediction of drug-target interaction networks from the integration of protein sequences and drug chemical structures. *Molecules*, 24(2):321, 2019. doi: 10.3390/molecules24020321. URL https://doi.org/10.3390/molecules24020321.

Wang, Y., Carter, B., Li, Z., and Huang, X. Application of machine learning methods in clinical trials for precision medicine. *JAMIA Open*, 5, 2021b. doi: 10.1093/jamiaopen/ooab107.

Wang, Z., Xiao, C., and Sun, J. Spot: Sequential predictive modeling of clinical trial outcome with meta-learning. In *Proceedings of the 14th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, 2023. doi: 10.1145/3584371.3613001.

Weber, A., Born, J., and Martínez, M. R. Titan: T-cell receptor specificity prediction with bimodal attention networks. *Bioinformatics*, 37:i237–i244, 2021. doi: 10.1093/bioinformatics/btab294.

Wolf, F. A., Angerer, P., and Theis, F. J. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biology*, 19:1–5, 2018.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. S. Moleculenet: A benchmark for molecular machine learning. *Chemical Science*, 9:513–530, 2018. doi: 10.1039/C7SC02664A.

Xu, M., Zhang, Z., Lu, J., Zhu, Z., Zhang, Y., Ma, C., Liu, R., and Tang, J. Peer: A comprehensive and multi-task benchmark for protein sequence understanding, 2022. Accepted by NeurIPS 2022 Dataset and Benchmark Track. arXiv v2: source code released; arXiv v1: release all benchmark results.

Yang, F., Wang, W., Wang, F., Fang, Y., Tang, D., Huang, J., Lu, H., and Yao, J. scbert as a large-scale pretrained deep language model for cell type annotation of single-cell rna-seq data. *Nature Machine Intelligence*, 4(10):852–866, 2022.

Yang, M., Huang, Z.-a., Zhou, W., Ji, J., Zhang, J., He, S., and Zhu, Z. Mix-tpi: a flexible prediction framework for tcr–pmhc interactions based on multimodal representations. *Bioinformatics*, 39, 2023. doi: 10.1093/bioinformatics/btad475.

Yaseen, A., Abbasi, W. A., and ul Amir Afsar Minhas, F. Protein binding affinity prediction using support vector regression and interfecial features. *2018 15th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, pp. 194–198, 2018. doi: 10.1109/IBCAST.2018.8312222.

Ye, X., Lee, Y. C., Gates, Z. P., Ling, Y., Mortensen, J. C., Yang, F. S., Lin, Y. S., and Pentelute, B. L. Binary combinatorial scanning reveals potent poly-alanine-substituted inhibitors of protein-protein interactions. *Communications Chemistry*, 5(1):128, Oct 2022. doi: 10.1038/s42004-022-00737-w.

You, Y. and Shen, Y. Cross-modality protein embedding for compound-protein affinity and contact prediction. *bioRxiv*, 2020. doi: 10.1101/2020.11.29.403162.

Yu, H. and Welch, J. D. Perturbnet predicts single-cell responses to unseen chemical and genetic perturbations. *bioRxiv*, 2022. doi: 10.1101/2022.07.20.500854. URL https://doi.org/10.1101/2022.07.20.500854.

Zambrano Chaves, J. M., Wang, E., Tu, T., Vaishnav, E. D., Lee, B., Mahdavi, S. S., Semturs, C., Fleet, D., Natarajan, V., and Azizi, S. Tx-llm: A large language model for therapeutics. *arXiv preprint arXiv:2406.06316*, 2024. doi: 10.48550/arXiv.2406.06316. URL https://arxiv.org/abs/2406.06316.

Zeng, J., Yu, G., Wang, J., Guo, M., and Zhang, X. Dmil-iii: Isoform-isoform interaction prediction using deep multi-instance learning method. *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 171–176, 2019. doi: 10.1109/BIBM47256.2019.8982956.

Zhang, Q., Yu, F. X., Wu, Y., et al. Novel gene therapy for rheumatoid arthritis with single local injection: adeno-associated virus-mediated delivery of a20/tnfaip3. *Military Medical Research*, 9(1):34, 2022. doi: 10.1186/s40779-022-00393-0. URL https://doi.org/10.1186/s40779-022-00393-0.

Zhang, S., Pyne, S., Pietrzak, S. J., Halberg, S., McCalla, S., Siahpirani, A. F., Sridharan, R., and Roy, S. Inference of cell type-specific gene regulatory networks on cell lineages from single cell omic datasets. *Nature Communications*, 14, 2023. doi: 10.1101/2022.07.25.501350.

Zhang, X., Xiao, C., Glass, L. M., and Sun, J. Deepenroll: patient-trial matching with deep embedding and entailment prediction. In *Proceedings of the web conference 2020*, pp. 1029–1037, 2020.

Zheng, S., Li, Y., Chen, S., Xu, J., and Yang, Y. Predicting drug–protein interaction using quasi-visual question answering system. *Nature Machine Intelligence*, 2:134–140, 2019. doi: 10.1038/s42256-020-0152-y.

Zhu, Y., Ouyang, X., Li, P., Jin, Q., Su, J., Zheng, L., and Ye, L. Torchdrug: A powerful and flexible machine learning platform for drug discovery. *Journal of Chemical Information and Modeling*, 62(9):2204–2212, 2022. doi: 10.1021/acs.jcim.2c00553.

Öztürk, H., Özgür, A., and Ozkirimli, E. Deepdta: deep drug–target binding affinity prediction. *Bioinformatics*, 34(17):i821–i829, Sep 2018. doi: 10.1093/bioinformatics/bty593.

A. New tasks: datasets, benchmarks, and code

PyTDC (Velez-Arce et al., 2024) introduces four tasks with fine-grained biological contexts: contextualized drug-target identification, single-cell chemical/genetic perturbation response prediction, and cell-type-specific protein-peptide binding interaction prediction, which introduce antigen-processing-pathway-specific, cell-type-specific, peptide-specific, and patient-specific biological contexts. Benchmarks for drug target nomination, genetic perturbation response prediction and chemical perturbation response prediction, all at single-cell resolution, were computed with corresponding leaderboards introduced on the TDC website. In addition, a benchmark and leaderboard was introduced for the TCR-Epitope binding interaction task for peptide design at single-cell resolution for T-cell receptors.

Context-specific metrics (section 3.3 and 4.2). In real-world machine learning applications, data subsets can correspond to critical outcomes. In therapeutics, there is evidence that the effects of drugs can vary depending on the type of cell they are targeting and where specific proteins are acting (Zhang et al., 2022). We build on the ”slice” abstraction (Chen et al., 2020a) to measure model performance at critical biological subsets. Context-specific metrics are defined to measure model performance at critical biological slices, with our benchmarks focused on measuring cell-type-specific model performance. In the case of benchmarks for perturbation response prediction and protein-peptide binding affinity, the studies are limited to a particular cell line, however our definition for context-specific metrics lays the foundations for building models which can generalize across cell lines and make context-aware predictions (Kwon et al., 2024). For single-cell drug-target nomination, we measure model performance at top-performing cell types. See corresponding sections 3.3 and 4.2 in the main text.

A.1. `tdc_ml.scDTI`: Contextualized Drug-Target Identification

Motivation. Single-cell data have enabled the study of gene expression and function at the level of individual cells across healthy and disease states (Jones et al., 2022; CZI Single-Cell Biology, et al., 2023; Kwon et al., 2024). To facilitate biological discoveries using single-cell data, machine-learning models have been developed to capture the complex, cell-type-specific behavior of genes (Theodoris et al., 2023; Cui et al., 2024; Yang et al., 2022; Li et al., 2024). In addition to providing the single-cell measurements and foundation models, PyTDC supports the development of contextual AI models to nominate therapeutic targets in a cell type-specific manner (Li et al., 2024). We introduce a benchmark dataset, model, and leaderboard for context-specific therapeutic target prioritization, encouraging the innovation of model architectures (e.g., to incorporate new modalities, such as protein structure and sequences (Li et al., 2020; 2019; Lee et al., 2018; Öztürk et al., 2018; Meng et al., 2017), genetic perturbation data (Ji et al., 2020; Ghadie et al., 2017; Zeng et al., 2019; Wang et al., 2021a), disease-specific single-cell atlases (Carlberg et al., 2019; Kuenzi et al., 2020; Julkunen et al., 2020), and protein networks (Parca et al., 2019; Zhang et al., 2023; Gupta et al., 2022)). PyTDC's release of `tdc_ml.scDTI` is a step in standardizing benchmarks for more comprehensive assessments of context-specific model performance.

A.1.1. Dataset and benchmark

We use curated therapeutic target labels from the Open Targets Platform (Targets, 2023) for rheumatoid arthritis (RA) and inflammatory bowel disease (IBD) (Li et al., 2024) (section A.1.2). We benchmark PINNACLE (Li et al., 2024)—trained on cell type specific protein-protein interaction networks—and a graph attention neural network (GAT) (Velickovic et al., 2017)—trained on a context-free reference protein-protein interaction network—on the curated therapeutic targets dataset. As expected, PINNACLE underperforms when evaluated on context-agnostic metrics (Table 3) and drastically outperforms GAT when evaluated on context-specific metrics (Appendix Table 4).

To our knowledge, PyTDC provides the first benchmark for context-specific learning (Kwon et al., 2024). PyTDC's contribution helps standardize the evaluation of single-cell ML models for drug target identification and other single-cell tasks (Yang et al., 2022; Theodoris et al., 2023; Li et al., 2024; Cui et al., 2024).

Model	AP@5	AUROC
PINNACLE (RA)	0.23	0.55
GAT (RA)	0.22	0.58
PINNACLE (IBD)	0.20	0.58
GAT (IBD)	0.20	0.64

Table 3. RA and IBD results for PINNACLE and a Graph Attention Network without cell-type-specific context metrics. These are results obtained after evaluating 10 seeds.

Table 4. **Cell-type specific target nomination for two therapeutic areas, rheumatoid arthritis (RA) and inflammatory bowel diseases (IBD).** Cell-type specific context metrics (definitions in Section 4.2): AP@5 Top-20 CT - average precision at $k = 5$ for the 20 best-performing cell types (CT); AUROC Top-1 CT - AUROC for top-performing cell type; AUROC Top-10 CT and AUROC Top-20 CT - weighted average AUROC for top-10 and top-20 performing cell types, respectively, each weighted by the number of samples in each cell type; AP@5/AUROC CF - context-free AP@5/AUROC integrated across all cell types. Shown are results from models run on ten independent seeds. N/A - not applicable.

Model	AP@5 Top-20 CT	AUROC Top-1 CT	AUROC Top-10 CT	AUROC Top-20 CT	AP@5 CF	AUROC CF
PINNACLE (RA)	0.913$\pm$0.059	0.765$\pm$0.054	0.676$\pm$0.017	0.647$\pm$0.014	0.226$\pm$0.023	0.510$\pm$0.005
GAT (RA)	N/A	N/A	N/A	N/A	0.220$\pm$0.013	0.580$\pm$0.010
PINNACLE (IBD)	0.873$\pm$0.069	0.935$\pm$0.067	0.799$\pm$0.017	0.752$\pm$0.011	0.198$\pm$0.013	0.500$\pm$0.010
GAT (IBD)	N/A	N/A	N/A	N/A	0.200$\pm$0.023	0.640$\pm$0.017

A.1.2. (Li, Michelle, et al.) Dataset

To curate target information for a therapeutic area, we examine the drugs indicated for the therapeutic area of interest and its descendants. The two therapeutic areas examined are rheumatoid arthritis (RA) and inflammatory bowel disease. For rheumatoid arthritis, we collected therapeutic data (i.e., targets of drugs indicated for the therapeutic area) from OpenTargets for rheumatoid arthritis (EFO 0000685), ankylosing spondylitis (EFO 0003898), and psoriatic arthritis (EFO 0003778). For inflammatory bowel disease, we collected therapeutic data for ulcerative colitis (EFO 0000729), collagenous colitis (EFO 1001293), colitis (EFO 0003872), proctitis (EFO 0005628), Crohn's colitis (EFO 0005622), lymphocytic colitis (EFO 1001294), Crohn's disease (EFO 0000384), microscopic colitis (EFO 1001295), inflammatory bowel disease (EFO 0003767), appendicitis (EFO 0007149), ulcerative proctosigmoiditis (EFO 1001223), and small bowel Crohn's disease (EFO 0005629).

We define positive examples (i.e., where the label y = 1) as proteins targeted by drugs that have at least completed phase 2 of clinical trials for treating a specific therapeutic area. As such, a protein is a promising candidate if a compound that targets the protein is safe for humans and effective for treating the disease. We retain positive training examples activated in at least one cell type-specific protein interaction network.

We define negative examples (i.e., where the label y = 0) as druggable proteins that do not have any known association with the therapeutic area of interest according to Open Targets. A protein is deemed druggable if targeted by at least one existing drug. We extract drugs and their nominal targets from Drugbank. We retain negative training examples activated in at least one cell type-specific protein interaction network.

Dataset statistics. The final number of positive (negative) samples for RA and IBD were 152 (1,465) and 114 (1,377), respectively. In (Li et al., 2024), this dataset was augmented to include 156 cell types.

Dataset split. Cold Split: We split the dataset such that about 80% of the proteins are in the training set, about 10% of the proteins are in the validation set, and about 10% of the proteins are in the test set. The data splits are consistent for each cell type context to avoid data leakage.

References. (Li et al., 2024)

Dataset license. CC BY 4.0

Code Sample. The dataset and splits are currently available on TDC Harvard Dataverse. In addition, you may obtain the protein splits used in (Li et al., 2024) via the following code.

```
from tdc_ml.resource.dataloader import DataLoader
data = DataLoader(name=opentargets_dti)
splits = data.get_split()
```

A.1.3. Code for datasets and benchmark

Retrieving training and validation datasets for the tdc_ml.scDTN task is as simple as 3 lines of code, with an additional 2 lines enabling model evaluation. Furthermore, PyTDC exposes the PPI network and cell types used for benchmarking of (Grover & Leskovec, 2016; Velickovic et al., 2017; Li et al., 2024; Brody et al., 2022) via a handful of lines of code as well.

Listing 1 The below code illustrates how to retrieve the train, test, and validation splits and run model evaluation for the tdc_ml.scDTN benchmark.

```
from tdc_ml.benchmark_group import scdti_group

group = scdtn_group.SCDTIGroup()
train, val = group.get_train_valid_split(seed=seed)  # Train model
test = group.get_test()["test"]  # Obtain predictions on the test set
group.evaluate(preds)  # Get benchmark results
```

Listing 2 The below code illustrates how to retrieve the PPI network and cell type labels (Li et al., 2024) used for training and benchmarking of models in tdc_ml.scDTN. Retrieval of cell type labels and embeddings from (Li et al., 2024) are also supported.

```
from tdc_ml.resource.pinnacle import PINNACLE
graph = pinnacle.get_ppi() # ppi network
embeds = pinnacle.get_embeds() # full pre-computed embeddings dataset
scproteins = pinnacle.get_keys() # protein cell types
```

A.1.4. `tdc_ml.scDTI`: Contextualized Drug-Target Nomination (Identification)

PyTDC introduces `tdc_ml.scDTI` task. The predictive, non-generative task is formalized as learning an estimator for a disease-specific function f of a target protein and cell type outputting whether the candidate protein t is a therapeutic target in that cell type c:

$$y = f(t, c). \tag{14}$$

Target candidate set. The target candidate set includes proteins, nucleic acids, or other molecules drugs can interact with, producing a therapeutic effect or causing a biological response. The target candidate set is constrained to proteins relevant to the disease being treated. It is denoted by:

$$\mathbb{T} = \{t_1, \ldots, t_{N_t}\}, \tag{15}$$

where $t_1, \ldots, t_{N_t}$ are N_t target candidates for the drugs treating the disease. Information modeled for target candidates can include interaction, structural, and sequence information.

Biological context set. The biological context set includes the cell-type-specific contexts in which the target candidate set operates. This set is denoted as:

$$\mathbb{C} = \{c_1, \ldots, c_{N_c}\}, \tag{16}$$

where $c_1, \ldots, c_{N_c}$ are N_c biological contexts on which drug-target interactions are being evaluated. Information modeled for cell-type-specific biological contexts can include gene expression and tissue hierarchy. The set is constrained to disease-specific cell types and tissues.

Drug-target identification. Drug-Target Identification is a binary label $y \in \{1, 0\}$, where $y = 1$ indicates the protein is a candidate therapeutic target. At the same time, 0 means the protein is not such a target.

The goal is to train a model f_θ for predicting the probability $\hat{y} \in [0, 1]$ that a protein is a candidate therapeutic target in a specific cell type. The model learns an estimator for a disease-specific function of a protein target $t \in \mathbb{T}$ and a cell-type-specific biological context $c \in \mathbb{C}$ as input, and the model is tasked to predict:

$$\hat{y} = f_\theta(t \in \mathbb{T}, c \in \mathbb{C}). \tag{17}$$

A.2. `tdc_ml.PerturbOutcome`: Perturbation-Response Prediction

Motivation. Understanding and predicting transcriptional responses to genetic or chemical perturbations provides insights into cellular adaptation and response mechanisms. Such predictions can advance therapeutic strategies, as they enable researchers to anticipate how cells will react to targeted interventions, potentially guiding more effective treatments. Models that have shown promise at this task (Roohani et al., 2023; Hetzel et al., 2022) are limited to either genetic or chemical perturbations without being able to generalize to the other. Approaches that can generalize across chemical and genetic perturbations (Piran et al., 2023; Yu & Welch, 2022) may be unable to generalize to unseen perturbations without modification.

A.2.1. Dataset and benchmark

We used the scPerturb (Peidli et al., 2024) datasets to benchmark the generalizability of perturbation-response prediction models across seen/unseen perturbations and cell lines. We benchmark models in genetic and chemical perturbations using metrics measuring intra/inter-cell line and seen/unseen perturbation generalizability. We provide results measuring unseen perturbation generalizability for Gene Perturbation Response Prediction using the scPerturb gene datasets (Norman K562, Replogle K562, Replogle RPE1) (Norman et al., 2019; Replogle et al., 2022) with results shown in Table 5. For Chemical Perturbation Prediction, we evaluated chemCPA utilizing cold splits on perturbation type and showed a significant decrease in performance for 3 of 4 perturbations evaluated in table (6). We have also included Biolord (Piran et al., 2023) and scGen (Lotfollahi et al., 2019) for comprehensive benchmarking on the well-explored perturbation response prediction on seen perturbation types problem. These tests were run on sciPlex2 (Srivatsan et al., 2019).

A.2.2. scPerturb Dataset

The scPerturb dataset is a comprehensive collection of single-cell perturbation data harmonized to facilitate the development and benchmarking of computational methods in systems biology. It includes various types of molecular readouts, such as transcriptomics, proteomics, and epigenomics. scPerturb is a harmonized dataset that compiles single-cell perturbation-response data. This dataset is designed to support the development and validation of computational tools by providing a consistent and comprehensive resource. The data includes responses to various genetic and chemical perturbations, crucial for understanding cellular mechanisms and developing therapeutic strategies. Data from different sources are uniformly pre-processed to ensure consistency. Rigorous quality control measures are applied to maintain high data quality. Features across different datasets are standardized for easy comparison and integration.

Dataset statistics. 44 publicly available single-cell perturbation-response datasets. Most datasets have, on average, approximately 3000 genes measured per cell. 100,000+ perturbations.

Dataset split. Cold Split and **Random Split** defined on cell lines and perturbation types.

References. (Peidli et al., 2024)

Dataset license. CC BY 4.0

Code Sample.

```
from tdc_ml.multi_pred.perturboutcome import PerturbOutcome
from pandas import DataFrame
test_loader = PerturbOutcome(
   name=scperturb_drug_AissaBenevolenskaya2021)
testdf = test_loader.get_data()
```

A.2.3. Genetic Perturbation Response Prediction

We use scPerturb gene datasets (Norman K562, Replogle K562, Replogle RPE1) (Norman et al., 2019; Replogle et al., 2022). In the case of single-gene perturbations, we assessed the models based on the perturbation of experimentally perturbed genes not included in the training data. We used data from two genetic perturbation screens, with 1,543 perturbations for RPE-1 (retinal pigment epithelium) cells and 1,092 for K-562 cells, each involving over 170,000 cells. These screens utilized the Perturb-seq assay, which combines pooled screening with single-cell RNA sequencing to analyze the entire transcriptome for each cell. We trained GEARS separately on each dataset. In addition to an existing deep learning-based model (CPA), we also developed a baseline model (no perturbation), assuming that gene expression remains unchanged after perturbation.

We evaluated the models' performance by calculating the mean squared error between the predicted gene expression after perturbation and the actual post-perturbation expression for the held-out set. Based on the highest absolute differential expression upon perturbation, the top 20 most differentially expressed genes were selected.

Table 5. **Unseen genetic perturbation response prediction.** We evaluate GEARS across the top 20 differentially expressed genes, based on the highest absolute differential expression upon perturbation, for MSE (MSE@20DEG). Gene expression was measured in log normalized counts. In single-cell analysis, a standard procedure is to normalize the counts within each cell so that they sum to a specific value (usually the median sum across all cells in the dataset) and then to log transform the values using the natural logarithm (Roohani et al., 2023). For both normalization and ranking genes by differential expression, we utilized Scanpy (Wolf et al., 2018). We used the sc.tl.rank_genes_groups() function with default parameters in Scanpy, which employs a t-test to estimate scores. This function provides a z-score for each gene and ranks genes based on the absolute values of the score. Genes showing a significant level of dropout were not included in this metric.

Dataset	Tissue	Cell Line	Method	MSE@20DEG
Norman K562	K562	lymphoblast	no-perturb	0.341±0.001
Norman K562	K562	lymphoblast	CPA	0.230±0.008
Norman K562	K562	lymphoblast	GEARS	0.176±0.003
Replogle 562	K562	lymphoblast	no-perturb	0.126±0.000
Replogle 562	K562	lymphoblast	CPA	0.126±0.000
Replogle 562	K562	lymphoblast	GEARS	0.109±0.004
Replogle RPE1	RPE-1	epithelial	no-perturb	0.164±0.000
Replogle RPE1	RPE-1	epithelial	CPA	0.162±0.001
Replogle RPE1	RPE-1	epithelial	GEARS	0.110±0.003

Listing 1. The below code illustrates how to retrieve the train, test, and val splits used for the tdc_ml.PerturbOutcome genetic perturbation benchmark

```python
from tdc_ml.benchmark_group import geneperturb_group
group = geneperturb_group.GenePerturbGroup()
train_val = group.get_train_valid_split()
test = group.get_test()
# train your model and test on the test set
group.evaluate(preds)
```

A.2.4. Chemical Perturbation Response Prediction

The dataset consists of four drug-based perturbations from sciPlex2 (Srivatsan et al., 2019; Peidli et al., 2024) (BMS, Dex, Nutlin, SAHA). sciPlex2 contains alveolar basal epithelial cells from the A549 (lung adenocarcinoma), K562 (chronic myelogenous leukemia), and MCF7 (mammary adenocarcinoma) tissues. Results are shown in Table 6. Our experiments rely on the coefficient of determination (R^2) as the primary performance measure. We calculate this score by comparing actual measurements with counterfactual predictions for all genes. Assessing all genes is essential to evaluating the decoder's overall performance and understanding the background context. Still, it is also beneficial to determine performance based on top differentially expressed genes (Hetzel et al., 2022). The baseline used discards all perturbation information, adequately measuring the improvement resulting by the models' drug encoding (Hetzel et al., 2022). ChemCPA's performance dropped by an average of 15% across the four perturbations. The maximum drop was 34%. Code for intra/inter cell-line benchmarks for chemical (drug) and genetic (CRISPR) perturbations is in Appendix A.2.4 and Appendix A.2.3, respectively. Using this code, users can evaluate models of their choice on the benchmark.

Table 6. **Unseen chemical perturbation response prediction.** We have evaluated chemCPA utilizing cold splits on perturbation type and show a significant decrease in performance for 3 of 4 perturbations evaluated. We have also included Biolord (Piran et al., 2023) and scGen (Lotfollahi et al., 2019) for comparison. The dataset used consists of four chemical (drug) perturbations from sciPlex2 (Peidli et al., 2024) (BMS, Dex, Nutlin, SAHA). sciPlex2 contains alveolar basal epithelial cells from the A549 (lung adenocarcinoma), K562 (chronic myelogenous leukemia), and MCF7 (mammary adenocarcinoma) tissues. Our experiments rely on the coefficient of determination (R^2) as the primary performance measure.

Drug	Method	R^2 (seen perturbations)	R^2 (unseen perturbations)
BMS	Baseline	0.620 ± 0.044	N/A
Dex	Baseline	0.603 ± 0.053	N/A
Nutlin	Baseline	0.628 ± 0.036	N/A
SAHA	Baseline	0.617 ± 0.027	N/A
BMS	Biolord	0.939 ± 0.022	N/A
Dex	Biolord	0.942 ± 0.028	N/A
Nutlin	Biolord	0.928 ± 0.026	N/A
SAHA	Biolord	0.980 ± 0.005	N/A
BMS	ChemCPA	0.943 ± 0.006	0.906 ± 0.006
Dex	ChemCPA	0.882 ± 0.014	0.540 ± 0.013
Nutlin	ChemCPA	0.925 ± 0.010	0.835 ± 0.009
SAHA	ChemCPA	0.825 ± 0.026	0.690 ± 0.021
BMS	scGen	0.903 ± 0.030	N/A
Dex	scGen	0.944 ± 0.018	N/A
Nutlin	scGen	0.891 ± 0.032	N/A
SAHA	scGen	0.948 ± 0.034	N/A

Listing 2. The below code illustrates how to retrieve the train, test, and val splits used for the tdc_ml.PerturbOutcome chemical perturbation benchmark

```
from tdc_ml.benchmark_group import counterfactual_group
group = counterfactual_group.CounterfactualGroup()
train, val = group.get_train_valid_split(remove_unseen=False)
test = group.get_test()
# train your model and test on the test set
group.evaluate(preds)
```

A.2.5. TDC_ML.PERTURBOUTCOME: PERTURBATION-RESPONSE PROBLEM FORMULATION

PyTDC introduces Perturbation-Response prediction task. The predictive, non-generative task is formalized as learning an estimator for a function of the cell-type-specific gene expression response to a chemical or genetic perturbation, taking a perturbation $p \in \mathbb{P}$, a pre-perturbation gene expression profile from the control set $e_0 \in \mathbb{E}_{\not{p}}$, and the biological context $c \in \mathbb{C}$ under which the gene expression response to the perturbation is being measured:

$$y = f(p, e_0, c). \tag{18}$$

We center our definition on regression for the cell-type-specific gene expression vector in response to a chemical or genetic perturbation.

Perturbation set. The perturbation set includes genetic and chemical perturbations. It is denoted by:

$$\mathbb{P} = \{p_1, \dots, p_{N_p}\}, \tag{19}$$

where $t_p, \dots, p_{N_p}$ are N_p evaluated perturbations. Information modeled for genetic perturbations can include the type of perturbation (i.e., knockout, knockdown, overexpression) and target gene(s) of the perturbation. Information modeled for chemical perturbations can include chemical structure (i.e., SMILES, InChl) and concentration and duration of treatment.

Control set. The control set includes the unperturbed gene expression profiles. This set is denoted as:

$$\mathbb{E}_{\not{p}} = \{\vec{e}_{0_1}, \dots, \vec{e}_{N_{e_0}}\}, \tag{20}$$

where $\vec{e}_{0_1}, \ldots, \vec{e}_{N_{e_0}}$ are N_{e_0} unperturbed gene expression profile vectors. Information models for gene expression profiles can include raw or normalized gene expression counts, transcriptomic profiles, and isoform-specific expression levels.

Biological context set. The biological context set includes the cell-type-specific contexts under which the perturbed gene expression profile is measured. It is denoted by:

$$\mathbb{C} = \{c_1, \ldots, c_{N_c}\}, \tag{21}$$

where $c_1, \ldots, c_{N_c}$ are the N_c biological contexts under which perturbations are being evaluated. Information modeled for biological contexts can include cell type or tissue type and experimental conditions (Peidli et al., 2024) as well as epigenetic markers (Agrawal et al., 2023; Nair et al., 2020).

Perturbation-response readouts. Perturbation-Response is a gene expression vector $\vec{e_1}$, where $\vec{e_1}_i$ denotes the expression of the i-th gene in the vector. It is the outcome of applying a perturbation, $p_i \in \mathbb{P}$, within a biological context, $c_j \in \mathbb{C}$, to a cell with a measured control gene expression vector, $e_{0_k} \in \mathbb{E}_{\not{\mathbb{P}}}$.

The Perturbation-Response Prediction learning task is to learn a regression model f_θ estimating the perturbation-response gene expression vector $\hat{\vec{e_1}}$ for a perturbation applied in a cell-type-specific biological context to a control:

$$\hat{\vec{e_1}} = f_\theta(p \in \mathbb{P}, e_0 \in \mathbb{E}_{\not{\mathbb{P}}}, c \in \mathbb{C}). \tag{22}$$

A.3. `tdc_ml.ProteinPeptide`: Contextualized Protein-Peptide Interaction Prediction

Motivation. Evaluating protein-peptide binding prediction models requires standardized benchmarks, presenting challenges in assessing and validating model performance across different studies (Romero-Molina et al., 2022). Despite the availability of several benchmarks for protein-protein interactions, this is not the case for protein-peptide interactions. The renowned multi-task benchmark for Protein sEquence undERstanding (PEER) (Xu et al., 2022) and MoleculeNet (Wu et al., 2018) both lack support for a protein-peptide interaction prediction task. Furthermore, protein-peptide binding mechanisms vary wildly by cellular and biological context (Janeway et al., 2001; Murphy & Weaver, 2016; Ploegh, 1998; Schatz & Swanson, 2011). Current models, as such, tend to be restricted to one task instance (i.e., T Cell Receptor (TCR) and Peptide-MHC Complex or B Cell Receptor (BCR) and Antigen Peptide binding) and do not span protein-peptide interactions (Springer et al., 2019; Chen et al., 2020b; Liu et al., 2021; Bradley, 2022; Hu & Liu, 2019; Lupia et al., 2019; Saxena et al., 2020).

We define and evaluate a subtask for TCR-Epitope binding interaction prediction applying contextual AI (Section A.3.5) to the T Cell cell line.

A.3.1. TCR-Epitope (Peptide-MHC Complex) Interaction Prediction

The critical challenge in TCR-Epitope (Peptide-MHC Complex) interaction prediction lies in creating a model that can effectively generalize to unseen TCRs and epitopes (Moris et al., 2020). While TCR-H (T et al., 2023) and TEINet (Jiang et al., 2023) have shown improved performance on prediction for known epitopes, by incorporating advanced features like attention mechanisms and transfer learning, the performance considerably drops for unseen epitopes (Cai et al., 2022; Weber et al., 2021). Another challenge in TCR-Epitope interaction prediction lies in the choice of heuristic for generating negative samples, with non-binders often underrepresented or biased in curated datasets, leading to inaccurate predictions when generalized (Kim et al., 2023).

Datasets and Benchmarks. PyTDC establishes a curated dataset and benchmark within its single-cell protein-peptide binding affinity prediction task to measure model generalizability to unseen TCRs and epitopes and model sensitivity to the selection of negative data points. Benchmarking datasets use three types of heuristics for generating negative samples: random shuffling of epitope and TCR sequences (RN), experimental negatives (NA), and pairing external TCR sequences with epitope sequences (ET). We harness data from the TC-hard dataset (Grazioli et al., 2022b) for the first two types and PanPep (Gao et al., 2023) for the third type. Both datasets use hard (Grazioli et al., 2022b) splits, ensuring that epitopes in the testing set are not present in the training set. Our results (Table 7) show the lack of a reasonable heuristic for generating negative samples, with model performance evaluation shown to be unsatisfactory. For two heuristics, all models perform poorly. The best-performing model in ET is MIX-TPI, with roughly 0.70 AUROC. The best-performing model in RN is AVIB-TCR, with approximately 0.576 AUROC. For NA, 4 of 6 models perform near-perfectly as measured on AUROC. Models benchmarked include AVIB-TCR (Grazioli et al., 2022a), MIX-TPI (Yang et al., 2023), Net-TCR2 (Jensen & Nielsen, 2023), PanPep (Moris et al., 2020), TEINet (Jiang et al., 2023), and TITAN (Weber et al., 2021).

Table 7. **TCR-epitope binding interaction binary classification performance.** All models perform poorly under realistic but challenging RN and ET experimental setups. The best-performing model in RN is AVIB-TCR, with an average of 0.576 (AUROC). The best-performing model in ET is MIX-TPI, with an average of 0.700 (AUROC). For NA, 4 of 6 models achieve near-perfect AUROC.

Methods	Experimental setup	ACC	F1	AUROC	AUPRC
AVIB-TCR	RN	0.570±0.028	0.468±0.086	0.576±0.049	0.605±0.044
MIX-TPI	RN	0.539±0.039	0.408±0.122	0.558±0.028	0.597±0.049
Net-TCR2	RN	0.528±0.050	0.354±0.036	0.551±0.042	0.554±0.075
PanPep	RN	0.507±0.028	0.473±0.039	0.535±0.021	0.579±0.040
TEINet	RN	0.459±0.036	0.619±0.036	0.535±0.029	0.581±0.043
TITAN	RN	0.476±0.063	0.338±0.111	0.502±0.066	0.523±0.055
AVIB-TCR	ET	0.611±0.012	0.553±0.020	0.683±0.010	0.815±0.006
MIX-TPI	ET	0.652±0.009	0.523±0.035	0.703±0.016	0.825±0.014
Net-TCR2	ET	0.621±0.027	0.522±0.020	0.674±0.017	0.810±0.016
PanPep	ET	0.556±0.009	0.506±0.011	0.638±0.009	0.753±0.009
TEINet	ET	0.356±0.008	0.512±0.010	0.571±0.009	0.646±0.011
TITAN	ET	0.670±0.013	0.492±0.048	0.624±0.021	0.733±0.018
AVIB-TCR	NA	0.636±0.062	0.197±0.169	0.944±0.021	0.949±0.023
MIX-TPI	NA	0.952±0.029	0.937±0.040	0.992±0.002	0.995±0.001
Net-TCR2	NA	0.655±0.051	0.274±0.123	0.973±0.009	0.985±0.005
PanPep	NA	0.419±0.011	0.352±0.006	0.611±0.014	0.499±0.031
TEINet	NA	0.413±0.023	0.582±0.023	0.973±0.011	0.981±0.006
TITAN	NA	0.695±0.050	0.404±0.141	0.629±0.053	0.661±0.040

A.3.2. TCHard Dataset

The TChard dataset is designed for TCR-peptide/-pMHC binding prediction. It includes over 500,000 samples from sources such as IEDB, VDJdb, McPAS-TCR, and the NetTCR-2.0 repository. The dataset is utilized to investigate how state-of-the-art deep learning models generalize to unseen peptides, ensuring that test samples include peptides not found in the training set. This approach highlights the challenges deep learning methods face in robustly predicting TCR recognition of peptides not previously encountered in training data.

Dataset statistics. 500,000 samples

Dataset split. Cold Split referred to as ”Hard” split in (Grazioli et al., 2022b).

References. (Grazioli et al., 2022b)

Dataset license. Non-Commercial Use

Code Sample.

```
from tdc_ml.resource.dataloader import DataLoader
data = DataLoader(name=tchard)
self.split = data.get_split()
```

A.3.3. PanPep Dataset

PanPep is a framework constructed in three levels for predicting the peptide and TCR binding recognition. We have provided the trained meta learner and external memory, and users can choose different settings based on their data available scenarios: Few known TCRs for a peptide: few-shot setting; No known TCRs for a peptide: zero-shot setting; plenty of known TCRs for a peptide: majority setting. More information is available in the Github repo .

Dataset statistics. Data from multiple studies involving millions of TCR sequences.

Dataset split. Cold Split referred to as ”Hard” split in (Gao et al., 2023).

References. (Gao et al., 2023)

Dataset license. GPL-3.0

Code Sample.

```
from tdc_ml.resource.dataloader import DataLoader
```

```
data = DataLoader(name=panpep)
self.split = data.get_split()
```

A.3.4. (Ye X et al) Dataset

Affinity selection-mass spectrometry data of discovered ligands against single biomolecular targets (MDM2, ACE2, 12ca5) from the Pentelute Lab of MIT This dataset contains affinity selection-mass spectrometry data of discovered ligands against single biomolecular targets. Several AS-MS-discovered ligands were taken forward for experimental validation to determine the binding affinity (KD) as measured by biolayer interferometry (BLI) to the listed target protein. If listed as a ”putative binder,” AS-MS alone was used to isolate the ligands to the target, with KD ¡ 1 uM required and often observed in orthogonal assays, though there is some (¡ 50%) chance that the ligand is nonspecific. Most of the ligands are putative binders, with 4446 total provided. For those characterized by BLI (only 34 total), the average KD is 266 ± 44 nM; the median KD is 9.4 nM.

Dataset statistics. 34 positive ligands, 4446 putative binders, and three proteins

Dataset Split. Stratified Split and **N/A Split**: We provide stratified 10/90 split on train/test as well as ”test set only” split.

References. (Ye et al., 2022; Brown et al., 2023)

Dataset license. CC BY 4.0

Code Sample.

```
from tdc_ml.multi_pred import ProteinPeptide
data = ProteinPeptide(name=brown_mdm2_ace2_12ca5)
data.get_split()
```

Listing 3. The below code illustrates how to retrieve the train, test, and val splits used for the tdc_ml.TCREpitope benchmark

```
from tdc_ml.benchmark_group.tcrepitope_group import TCREpitopeGroup
group = TCREpitopeGroup()
train_val = group.get_train_valid_split()
test = group.get_test()
# train your model and test on the test set
group.evaluate(preds)
```

A.3.5. tdc_ml.ProteinPeptide: Protein-Peptide Interaction Prediction Problem Formulation

PyTDC introduces the Protein-Peptide Binding Affinity prediction task. The predictive, non-generative task is to learn a model estimating a function of a protein, peptide, antigen processing pathway, biological context, and interaction features. It outputs a binding affinity value (e.g., dissociation constant Kd, Gibbs free energy ΔG) or binary label indicating strong or weak binding. The binary label can also include additional biomarkers, such as allowing for a positive label if and only if the binding interaction is specific (Brown et al., 2023; Hoofnagle & Resing, 2001; Smith & Kelleher, 2013). To account for additional biomarkers beyond binding affinity value, our task is specified with a binary label.

Protein set. The protein set includes target proteins. It is denoted by:

$$\mathbb{P} = \{p_1, \ldots, p_{N_p}\}, \tag{23}$$

where $p_1, \ldots, p_{N_p}$ are N_p target proteins. Information modeled for proteins can include sequence, structural, or post-translational modification data.

Peptide set. The control set includes the peptide candidates. This set is denoted as:

$$\mathbb{S} = \{s_1, \ldots, s_{N_s}\}, \tag{24}$$

where $s_1, \ldots, s_{N_s}$ are N_s candidate peptides. Information modeled for candidate peptides can include sequence, structural, and physicochemical data.

Antigen processing pathway set. The antigen processing pathway set includes antigen processing pathway profile information about prior steps in the biological antigen presentation pathway processes. It is denoted by:

$$\mathbb{A} = \{a_1, \ldots, a_{N_a}\}, \tag{25}$$

where $a_1, \ldots, a_{N_a}$ are the N_a antigen processing pathway profiles modeled. Information modeled in a profile can include proteasomal cleavage sites (Nielsen et al., 2005), classification into viral, bacterial, and self-protein sources and endogenous vs exogenous processing pathway (Reynisson et al., 2020; O'Donnell et al., 2020; Saxena et al., 2020; Boulanger et al., 2018), and target/receptor-specific pathway attributes such as transporter associated with antigen processing (TAP) affinity (Bhasin et al., 2007), and endosomal/lysosomal processing efficiency (Koşaloğlu-Yalçın et al., 2020).

Interaction set. It contains the interaction feature profiles. The set is denoted by:

$$\mathbb{I} = \{i_1, \ldots, i_{N_i}\}, \tag{26}$$

where $i_1, \ldots, i_{N_i}$ are the N_i interaction feature profiles. Information modeled in an interaction feature profile can include contact maps (Huang & Ding, 2022; Yaseen et al., 2018; Li et al., 2021; You & Shen, 2020), distance maps (Yaseen et al., 2018; Zheng et al., 2019), electrostatic interactions (Huang & Ding, 2022), and hydrogen bonds (Huang & Ding, 2022).

Cell-type-specific biological context set. It contains the interaction feature profiles. The set is denoted by:

$$\mathbb{C} = \{c_1, \ldots, c_{N_c}\}, \tag{27}$$

where $c_1, \ldots, c_{N_c}$ are the N_c cell-type-specific biological contexts under which the protein-peptide interaction is being evaluated. Information modeled in the cell-type-specific biological context can include transcriptomic and proteomic data. We note, however, that, to our knowledge, single-cell transcriptomic and proteomic data has yet to be used in protein-peptide binding affinity prediction, outlining a promising avenue of research in developing machine learning models for peptide-based therapeutics.

Protein-peptide interaction. It is a binary label, $y \in \{1, 0\}$, where $y = 1$ indicates a protein-peptide pair met the target biomarkers and $y = 0$ indicates the pair did not meet the target biomarkers.

The Protein-Peptide Interaction Prediction learning task is to learn a binary classification model f_θ estimating the probability, $\hat{y}$, of a protein-peptide interaction meeting specific biomarkers:

$$\hat{y} = f_\theta(p \in \mathbb{P}, s \in \mathbb{S}, a \in \mathbb{A}, i \in \mathbb{I}, c \in \mathbb{C}). \tag{28}$$

A.4. tdc_ml.TrialOutcome

PyTDC introduces a model framework, task definition, dataset, and benchmark for the Clinical Outcome Prediction task tailored to precision medicine. The framework and definition aim to assess clinical trials systematically and comprehensively by predicting various endpoints for patient sub-populations. Our benchmark uses the Trial Outcome Prediction (TOP) dataset (Fu et al., 2022). TOP consists of 17,538 clinical trials with 13,880 small-molecule drugs and 5,335 diseases.

Dataset and benchmark. Our benchmark uses the Trial Outcome Prediction (TOP) dataset (Fu et al., 2022). TOP consists of 17,538 clinical trials with 13,880 small-molecule drugs and 5,335 diseases. Out of these trials, 9,999 (57.0%) succeeded (i.e., meeting primary endpoints), and 7,539 (43.0%) failed. Out of these trials, 1,787 were in Phase I testing (toxicity and side effects), 6,102 in Phase II (efficacy), and 4,576 in Phase III (effectiveness compared to current standards). We perform a temporal split for benchmarking. The train/validation and test are time-split by the date January 1, 2014, i.e., the start dates of the test set are after January 1, 2014, while the completion dates of the train/validation set are before January 1, 2014. Here, the HINT model (Fu et al., 2022), is benchmarked against COMPOSE (Gao et al., 2020) and DeepEnroll (Zhang et al., 2020) models. Results are shown in table 8.

Table 8. Clinical Trial Outcome Prediction task benchmark model results on the TOP dataset (Fu et al., 2022), described in section A.4 (Velez-Arce et al., 2024).

Model	Phase 1 AUPRC	Phase 2 AUPRC	Phase 3 AUPRC	Indication Level AUPRC
HINT	0.772	0.607	0.623	0.703
COMPOSE	0.665	0.532	0.545	0.624
DeepEnroll	0.701	0.580	0.590	0.655

A.4.1. TOP Dataset

TOP (Fu et al., 2022) consists of 17,538 clinical trials with 13,880 small-molecule drugs and 5,335 diseases. Out of these trials, 9,999 (57.0%) succeeded (i.e., meeting primary endpoints), and 7,539 (43.0%) failed. For each clinical trial, we produce the following four data items: (1) drug molecule information, including Simplified Molecular Input Line Entry System (SMILES) strings and molecular graphs for the drug candidates used in the trials; (2) disease information including ICD-10 codes (disease code), disease description, and disease hierarchy in terms of CCS codes (https://www.hcup-us.ahrq.gov/toolssoftware/ccs10/ccs10.jsp); (3) trial eligibility criteria are in unstructured natural language and contain inclusion and exclusion criteria; and (4) trial outcome information includes a binary indicator of trial success (1) or failure (0), trial phase, start and end date, sponsor, and trial size (i.e., number of participants).

Dataset statistics. Phase I: 2,402 trials / Phase II: 7,790 trials / Phase III: 5,741 trials.

Dataset split. Temporal Split as defined in (Fu et al., 2022) and Section A.4.

References. (Fu et al., 2022)

Dataset license. Non-Commercial Use

Code Sample.

```
from tdc_ml.multi_pred import TrialOutcome
data = TrialOutcome(name = 'phase1') # 'phase2' / 'phase3'
split = data.get_split()
```

A.4.2. Clinical Trial Outcome Prediction Problem Formulation

The Clinical Trial Outcome Prediction task is formulated as a binary classification problem, where the machine learning model predicts whether a clinical trial will have a positive or negative outcome. It is a function that takes patient data, trial design, treatment characteristics, disease, and macro variables as inputs and outputs a trial outcome prediction, a binary indicator of trial success (1) or failure (0).

Patient set. The patient set includes one or multiple patient sub-populations, with the extreme case representing personalization. It is denoted as follows:

$$\mathbb{P} = \{p_1, \dots, p_{N_p}\}, \tag{29}$$

where $p_1, \dots, p_{N_p}$ are N_p patient sub-populations in this trial. The TOP benchmark (Fu et al., 2022) dataset represents patient data as part of the trial eligibility criteria. Patient data can include demographics (Hong et al., 2021; Lv et al., 2020; Rongali et al., 2020; Rahimian et al., 2018; Park et al., 2020), baseline health metrics (Rahimian et al., 2018; Park et al., 2020; Bedon et al., 2021), and medical history (Hong et al., 2021; Lv et al., 2020; Rongali et al., 2020; Rahimian et al., 2018; Park et al., 2020).

Trial design set. The trial design set includes this clinical trial's design profiles. It is denoted as:

$$\mathbb{D} = \{d_1, \dots, d_{N_d}\}, \tag{30}$$

where $d_1, \dots, d_{N_d}$ are N_d eligible trial design profiles for this clinical trial. Trial design profiles can model information including phase of the trial (Fu et al., 2022), number of participants, duration of the trial, trial eligibility criteria (Fu et al., 2022), and randomization and blinding methods (Schperberg et al., 2020; Wang et al., 2021b; Dai et al., 2022).

Treatment set. The treatment set includes the candidate treatments for the trial. It is denoted as:

$$\mathbb{T} = \{t_1, \dots, t_{N_t}\}, \tag{31}$$

where $t_1, \dots, t_{N_t}$ are N_t candidate treatments for the clinical trial. The information modeled for treatments can include type of treatment (drug (Fu et al., 2022; Brbi et al., 2024), device (Kalscheur et al., 2018; Fujima et al., 2019; van Os et al., 2018), procedure (Asadi et al., 2014; Arvind et al., 2018; Senders et al., 2018; Jourahmad et al., 2023; MacKay et al., 2021)), dosage and administration route (Schperberg et al., 2020; Bedon et al., 2021; Bowman et al., 2023), mechanism of action (Sayaman et al., 2020; Beacher et al., 2021; Siah et al., 2019), pre-clinical and early-phase trial results (Beacher et al., 2021; Bedon et al., 2021; Beinse et al., 2019; Wang et al., 2023).

Macro context set. The macro context set contains the configurations of macro variables relevant to the clinical trial. It is denoted as:

$$\mathbb{C} = \{c_1, \dots, c_{N_c}\}, \tag{32}$$

where $c_1, \ldots, c_{N_c}$ are N_c configurations containing the values for macro variables relevant to the trial, which can include geography (Shamout et al., 2020; Beacher et al., 2021; Wang et al., 2023; Liu & Salinas, 2017) and regulatory considerations (Beacher et al., 2021; Shamout et al., 2020).

Trial outcomes. The trial outcome is a binary label $y \in \{1, 0\}$, where $y = 1$ indicates the trial met their primary endpoints, while 0 means failing to meet with the primary endpoints.

The learning task is to learn a model f_θ for predicting the trial success probability $\hat{y}$, where $\hat{y} \in [0, 1]$:

$$\hat{y} = f_\theta(p \in \mathbb{P}, d \in \mathbb{D}, t \in \mathbb{T}, c \in \mathbb{C}). \tag{33}$$

A.5. tdc_ml.sBDD Structure-Based Drug Design

Structure-based drug design aims to create diverse new molecules that bind to protein pockets (3D structures) and have favorable chemical properties. These attributes are evaluated using pharmaceutically-relevant oracle functions. In this task, an ML model learns molecular traits of protein pockets from a comprehensive dataset of protein-ligand pairs. Subsequently, potential new molecules can be generated using the acquired conditional distribution. The generated molecules must exhibit outstanding properties, including high binding effectiveness and structural variety. They must meet other user-specified criteria, such as the feasibility of synthesis (synthesizability/designability) and similarity to known drugs.

A.5.1. PDBBind Dataset

PDBBind is a comprehensive database extracted from PDB with experimentally measured binding affinity data for protein-ligand complexes. PDBBind does not allow the dataset to be re-distributed in any format. Thus, we could not host it on the TDC server. However, we provide an alternative route since significant processing is required to prepare the dataset ML. The user only needs to register at http://www.pdbbind.org.cn/, download the raw dataset, and then provide the local path. TDC will then automatically detect the path and transform it into an ML-ready format for the TDC data loader.

Dataset statistics. 19,445 protein-ligand pairs

Dataset split. Random Split

References. (Liu et al., 2015)

Dataset license. See note in the description on the TDC website.

Code Sample.

```python
from tdc_ml.generation import SBDD
data = SBDD(name='PDBBind', path='./pdbbind')
split = data.get_split()
```

A.5.2. DUD-E Dataset

DUD-E provides a directory of valuable decoys for protein-ligand docking.

Dataset statistics. 22,886 active compounds and affinities against 102 targets. DUD-E does not support pocket extraction as protein and ligand are not aligned.

Dataset split. Random Split

References. (Mysinger et al., 2012)

Dataset license. Not specified

Code Sample.

```python
from tdc_ml.generation import SBDD
data = SBDD(name='dude')
split = data.get_split()
```

A.5.3. scPDB Dataset

scPDB is processed from PDB for structure-based drug design that identifies suitable binding sites for protein-ligand docking.

Dataset statistics. 16,034 protein-ligand pairs over 4,782 proteins and 6,326 ligands

Dataset split. Random Split

References. (Meslamani et al., 2011)

Dataset license. Not specified

Code Sample.

```
from tdc_ml.generation import SBDD
data = SBDD(name='scPDB')
split = data.get_split()
```

A.5.4. Structure-Based Drug Design Problem Formulation

Structure-based Drug Design aims to generate diverse, novel molecules with high binding affinity to protein pockets (3D structures) and desirable chemical properties. Oracle functions measure these properties. A machine learning task first learns the molecular characteristics given specific protein pockets from a large set of protein-ligand pair data. Then, from the learned conditional distribution, we can sample novel candidates.

Target candidate set. The target candidate set includes proteins, nucleic acids, or other biomolecules drugs can interact with, producing a therapeutic effect or causing a biological response. It is denoted by:

$$\mathbb{T} = \{t_1, \dots, t_{N_t}\}, \tag{34}$$

where $t_1, \dots, t_{N_t}$ are N_t target candidates for the evaluated set of drugs. Information modeled for target candidates can include interaction, structural, and sequence information.

Ligand candidate set. The ligand drug candidate set includes the drug molecules being tested for a particular therapeutic effect or biological response. It is denoted by:

$$\mathbb{L} = \{l_1, \dots, l_{N_l}\}, \tag{35}$$

where $l_1, \dots, l_{N_l}$ are the N_l ligand/drug molecules being evaluated. Drug modeling can include molecular structure, often represented in formats such as SMILES (Simplified Molecular Input Line Entry System) or InChI (International Chemical Identifier) (Öztürk et al., 2019), physicochemical properties like hydrophobicity and molecular weight (Chen et al., 2020b), and molecular descriptors and fingerprints (Wang et al., 2019).

Scoring function. The scoring function, denoted by S, evaluates the binding affinity of ligand $l \in \mathbb{L}$ to protein target $t \in \mathbb{T}$.

Drug-likeness function. Function representing the drug-likeness of ligand $l \in \mathbb{L}$, including properties like solubility, stability, and toxicity.

The generative learning task is to generate the ligand $l \in \mathbb{L}$ maximizing binding affinity, S_θ, and drug-likeness, f_θ. Given a loss function, $\text{Loss}(S(t,l), f(l))$, for $t \in \mathbb{T}$ and $l \in \mathbb{L}$, the first step is to learn a model M_θ s.t.,

$$M_\theta = \text{argmin}_\theta[\text{Loss}(S_\theta(t,l), f_\theta(l))]. \tag{36}$$

This is followed by the ligand optimization step, which optimizes the ligand for maximum binding affinity and drug-likeness given the trained model. A ligand optimization function, F, such as addition or multiplication, is used for the optimization:

$$l^* = \text{argmax}_{l \in \mathbb{L}}[F(S_\theta(t \in \mathbb{T}, l), f_\theta(l))]. \tag{37}$$

An example formulation would be as follows:

$$l^* = \text{argmax}_{l \in \mathbb{L}}[S_\theta(t \in \mathbb{T}, l) \times f_\theta(l)]. \tag{38}$$

B. PyTDC Architecture

B.1. API-First Design and Model-View-Controller

See section 3.1 in the main text. PyTDC (Velez-Arce et al., 2024) drastically expands dataset retrieval capabilities available in TDC-1 beyond those of other leading benchmarks. Leading benchmarks, like MoleculeNet (Wu et al., 2018) and

TorchDrug (Zhu et al., 2022) have traditionally provided dataloaders to access file dumps. PyTDC introduces API-integrated multimodal data-views (Churi et al., 2016; Emmerik et al., 1993; Rammerstorfer & Mössenböck, 2003). To do so, the software architecture of PyTDC was redesigned using the Model-View-Controller (MVC) design pattern (Bucanek, 2009; Malik et al., 2021). The MVC architecture separates the model (data logic), view (UI logic), and controller (input logic), which allows for the integration of heterogeneous data sources and ensures consistency in data views (Churi et al., 2016). The MVC pattern supports the integration of multiple data modalities by using data mappings and views (Rammerstorfer & Mössenböck, 2003). The MVC-enabled-multimodal retrieval API is powered by PyTDC's Resource Model (Section B.2).

B.1.1. TDC DataLoader (*Model*)

As per the TDC-1 specification, this component queries the underlying data source to provide raw or processed data to upstream function calls. We augmented this component beyond TDC-1 functionality to allow for querying datasets introduced in PyTDC, such as the CZ CellXGene.

B.1.2. TDC meaningful data splits and multimodal data processing (*View*)

As per the TDC-1 specification, this component implements data splits to evaluate model generalizability to out-of-distribution samples and data processing functions for multiple modalities. We augmented this component to act on data views (Churi et al., 2016) specified by PyTDC's controller.

B.1.3. PyTDC Domain-Specific Language (*Controller*)

PyTDC develops an Application-Embedded Domain-Specific Data Definition Programming Language facilitating the integration of multiple modalities by generating data views from a mapping of multiple datasets and functions for transformations, integration, and multimodal enhancements, while mantaining a high level of abstraction (Membarth et al., 2016) for the Resource framework. We include examples developing multimodal datasets leveraging this MVC DSL in listing 4.

B.2. Resource Model

The Commons introduces a redesign of TDC-1's dataset layer into a new data model dubbed the PyTDC resource, which has been developed under the MVC paradigm to integrate multiple modalities into the API-first model of PyTDC.

B.2.1. CZ CellXGene with single cell biology datasets

CZ CellXGene (CZI Single-Cell Biology, et al., 2023) is a open-source platform for analysis of single-cell RNA sequencing data. We leverage the CZ CellXGene to develop a PyTDC Resource Model for constructing large-scale single-cell datasets that maps gene expression profiles of individual cells across tissues, healthy and disease states. PyTDC leverages the SOMA (Stack of Matrices, Annotated) API, adopts TileDB-SOMA (Papadopoulos et al., 2016) for modeling sets of 2D annotated matrices with measurements of features across observations, and enables memory-efficient querying of single-cell modalities (i.e., scRNA-seq, snRNA-seq), across healthy and diseased samples, with tabular annotations of cells, samples, and patients the samples come from.

We develop a remote procedure call (RPC) API taking the string name (e.g., listing 5 in section B.3.1) of the desired reference dataset as specified in the CellXGene (CZI Single-Cell Biology, et al., 2023). The remote procedure call for fetching data is specified as a Python generator expression, allowing the user to iterate over the constructed single-cell atlas without loading it into memory (Tasoluk & Tanrikulu, 2023). Specifying the RPC as a Python generator expression allows us to make use of memory-efficient querying as provided by TileDB (Papadopoulos et al., 2016). The single cell datasets can be integrated with therapeutics ML workflows in PyTDC by using tools such as PyTorch's IterableDataset module (Paszke et al., 2019).

Knowledge graph, external APIs, and model hub. We have developed a framework for biomedical knowledge graphs to enhance multimodality of dataset retrieval via PyTDC's Resource Model. Our system leverages PrimeKG to integrate 20 high-quality resources to describe 17,080 diseases with 4,050,249 relationships (Chandak et al., 2023). Our framework also extends to external APIs, with data views currently leveraging BioPython (Cock et al., 2009), for obtaining nucleotide sequence information for a given non-coding RNA ID from NCBI (Cock et al., 2009), and The Uniprot Consortium's RESTful GET API (Consortium, 2021) for obtaining amino acid sequences. In addition we've developed the framework to allow access to embedding models under diverse biological contexts via the PyTDC Model Hub. Examples using these components are in sections B.3.2 and B.3.3.

B.3. API-first design code samples

We provide code samples for the components described in sections 3.1, B.1, and B.2.

Listing 4. The above configuration augments a protein-peptide dataset with an additional modality, amino acid sequence, and invokes numerous data processing functions tailored to the specific needs of the underlying dataset. Added information for this demonstration can be found at: . There are more complex workflows implemented for current TDC dataviews and all such views leveraging the DSL can be found in the repo at

```
from .config import DatasetConfig
from ..feature_generators.protein_feature_generator import
    ↪ ProteinFeatureGenerator


class BrownProteinPeptideConfig(DatasetConfig):
   Configuration␣for␣the␣brown-protein-peptide␣datasets

   def __init__(self):
      super(BrownProteinPeptideConfig, self).__init__(
         dataset_name=brown_mdm2_ace2_12ca5,
         data_processing_class=ProteinFeatureGenerator,
         functions_to_run=[
            autofill_identifier, create_range, insert_protein_sequence
         ],
         args_for_functions=[{
            autofill_column: Name,
            key_column: Sequence,
         }, {
            column: KD␣(nM),
            keys: [Putative␣binder],
            subs: [0]
         }, {
            gene_column: Protein␣Target
         }],
         var_map={
            X1: Sequence,
            X2: protein_or_rna_sequence,
            ID1: Name,
            ID2: Protein␣Target,
         },
      )
```

B.3.1. PyTDC Multimodal Single-Cell Retrieval API

We focus on the use case of an ML researcher who wishes to train a model on a large-scale single-cell atlas. In particular, researchers would be familiar with and have trained models on traditional single-cell datasets such as Tabula Sapiens (Jones et al., 2022). Their interest is to scale a model by training it on a more extensive single-cell atlas based on this reference dataset. We build such an API. Specifically, given a reference dataset available in CellXGene Discover (CZI Single-Cell Biology, et al., 2023), we allow the user to perform a memory-efficient query using TileDB-SOMA to expand the reference dataset to include cell entries with non-zero readouts for any of the genes present in the reference dataset. This allows users to build large-scale single-cell atlases on familiar reference datasets. The example below illustrates how a user may construct a large-scale atlas with Tabula Sapiens as the reference dataset. Other use cases include augmenting datasets using knowledge graphs and cell-type-specific biomedical contexts. These capabilities are all powered by the

Model-View-Controller framework (section B.1).

Listing 5. The example below illustrates how a user may construct a large-scale atlas with Tabula Sapiens as the reference dataset using the PyTDC CELLXGENE API.

```
from tdc_ml.multi_pred.single_cell import CellXGene
dataloader = CellXGene(name=Tabula␣Sapiens␣-␣All␣Cells)
gen = dataloader.get_data(
   value_filter=tissue␣==␣'brain'␣and␣sex␣==␣'male'
)
df = next(gen)
```

Listing 6. In addition to our PyTDC DataLoader API implementation for the CellXGene RPC API, we provide a simplified wrapper over the CellXGene Census Discovery API, which allows users to perform remote procedure calls to fetch Cell Census data in more machine-learning-friendly formats like Pandas and Scipy. We also maintain support for the AnnData format. Users can query Cell Census counts as well as metadata using this API. The code sample below illustrates such usage.

```
from tdc_ml.resource import cellxgene_census

# initialize Census Resource and query filters
resource = cellxgene_census.CensusResource()
cell_value_filter = tissue␣==␣'brain'␣and␣sex␣==␣'male'
cell_column_names = [assay, cell_type, tissue]

# Obtaining cell metadata from the cellxgene census in pandas format
obsdf = resource.get_cell_metadata(
   value_filter=cell_value_filter,
   column_names=cell_column_names,
   fmt=pandas)
```

B.3.2. PrimeKG Knowledge Graph

PrimeKG supports drug-disease prediction by including an abundance of 'indications,' 'contradictions', and 'off-label use' edges, which are usually missing in other knowledge graphs. We accompany PrimeKG's graph structure with text descriptions of clinical guidelines for drugs and diseases to enable multimodal analyses (Chandak et al., 2023). The code below depict example use cases of the PyTDC PrimeKG API. Demonstrations are additionally available in https://colab.research.google.com/drive/1kYH8nt3nW7tXYBPNcfYuDbWxGTqOEnWg?usp=sharing.

Listing 7. We illustrate here example utilities for retrieving drug-target-disease associations using the PyTDC PrimeKG API

```
from tdc_ml.resource import PrimeKG

pkg = PrimeKG()
pkgdf = pkg.get_data()

def get_all_drug_evidence(disease):
   given␣a␣disease,␣retrieve␣all␣drugs␣interacting␣with␣proteins␣relevant␣to␣
       ↪ disease
   prots = pkgdf[(pkgdf[relation] == disease_protein) & (pkgdf[x_name] == disease
       ↪ )][y_name].unique()
   drugs = pkgdf[(pkgdf[relation] == drug_protein) & (pkgdf[y_name].isin(prots))]
   relations = drugs[display_relation].unique()
   out = {}
   for rel in relations:
      out[rel] = drugs[drugs[display_relation] == rel][x_name].unique()
```

```
    return out

def get_all_associated_targets(disease):
    return pkgdf[(pkgdf[relation] == disease_protein) & (pkgdf[x_name] == disease)
        ↪ ][[y_name, display_relation]]

def get_disease_disease_associations(disease):
    return pkgdf[(pkgdf[relation] == disease_disease) & (pkgdf[x_name] == disease)
        ↪ ][[y_name, display_relation]]

def get_labels_from_evidence(disease):
    diseases = get_disease_disease_associations(disease)[y_name]
    out = set()
    for d in diseases:
        targets = get_all_associated_targets(d)[y_name].unique()
        out.update(targets)
    return list(out)

def all_diseases_by_keyword(kw):
    return pkgdf[(pkgdf[relation] == disease_protein) & (pkgdf[x_name].str.
        ↪ contains(kw, case=False, na=False))][x_name].unique()

if __name__ == __main__:
    x = all_diseases_by_keyword(autism)
    [get_all_drug_evidence(d) for d in x]
    [get_all_associated_targets(d) for d in x]
    [get_disease_disease_associations(d) for d in x]
    print([get_labels_from_evidence(d) for d in x])
```

Listing 8. Here we illustrate combinig the PyTDC PrimeKG API with the networkx module to retrieve drug repositioning opportunities.

```
import networkx as nx
from tdc_ml.resource import PrimeKG

# Load the PrimeKG data
kg = PrimeKG()
data = kg.get_data()
data = data[data[relation].str.contains(drug)]

# Create a graph from the knowledge graph data
G = nx.from_pandas_edgelist(data, 'x_id', 'y_name', edge_attr='relation')

# Example function to find repositioning opportunities for a given drug
def find_repositioning_opportunities(drug):
    neighbors = list(G.neighbors(drug))
    diseases = [node for node in neighbors if G[drug][node]['relation'] == '
        ↪ drug_protein']
    return diseases

# Find repositioning opportunities for a specific drug
drug_name = 'DB00945'
repositioning_opportunities = find_repositioning_opportunities(drug_name)
```

B.3.3. PyTDC Model Hub

The introduced model hub is composed of the PyTDC Model Hub and a set of utilities and endpoints for facilitating model inference and fine-tuning. PyTDC introduces The Commons' HuggingFace Model Hub. It is a resource with pre-trained models, including geometric deep learning models, large language models, and other contextualized multimodal models for therapeutic tasks. The models can be fine-tuned using datasets in PyTDC and be used for downstream tasks such as implementations of multi-agent collaborative schemes (Gao et al., 2024) (i.e., expert consultants) our predictive therapeutic tasks (Theodoris et al., 2023; Initiative, 2023). The model hub details and available models can be found at https://huggingface.co/tdc.

Listing 9. The below illustrates the basic functionality of the model hub to download a model and perform inference on a precdictive task as well as fine-tune the model

```
from tdc_ml import tdc_hf_interface
tdc_hf = tdc_hf_interface(BBB_Martins-AttentiveFP)
# load deeppurpose model from this repo
dp_model = tdc_hf.load_deeppurpose('./data')
tdc_hf.predict_deeppurpose(dp_model, ['YOUR␣SMILES␣STRING'])
# fine-tune
dp_model.train(train, val, test) # for some defined splits
```

Listing 10. The below illustrates using the tdc model hub to download a foundation model (Theodoris et al., 2023)

```
from tdc_ml import tdc_hf_interface
from transformers import BertModel
geneformer = tdc_hf_interface(Geneformer)
model = geneformer.load()
assert isinstance(model, BertModel), type(model)
```

Listing 11. Beyond downloading a foundation model (Theodoris et al., 2023), the model server facilitates model inference across a range of datasets. Below an example integrating the PyTDC CellXGene API with the model server.

```
from tdc_ml.resource import cellxgene_census
from tdc_ml.model_server.tokenizers.geneformer import GeneformerTokenizer
from tdc import tdc_hf_interface
import torch

# query the CELLXGENE census
adata = self.resource.get_anndata(
var_value_filter=
feature_id␣in␣['ENSG00000161798',␣'ENSG00000188229'],
obs_value_filter=
sex␣==␣'female'␣and␣cell_type␣in␣['microglial␣cell',␣'neuron'],
column_names={
   obs: [
      assay, cell_type, tissue, tissue_general,
      suspension_type, disease
   ]
},
)

# tokenize gene expression vectors
tokenizer = GeneformerTokenizer()
x = tokenizer.tokenize_cell_vectors(adata,
                        ensembl_id=feature_id,
                              ncounts=n_measured_vars)

cells, _ = x

# load the model
geneformer = tdc_hf_interface(Geneformer)
model = geneformer.load()


Custom␣pre-processing␣code␣can␣include␣padding␣and␣attention␣mask␣definitions.


input_tensor = torch.tensor(cells)
out = []
for batch in input_tensor:
    # build an attention mask
    attention_mask = torch.tensor(
        [[x[0] != 0, x[1] != 0] for x in batch])
    # run batched inference
    out.append(model(batch, attention_mask=attention_mask))
```

C. PyTDC model server: models and code

We have released open source model retrieval and deployment software that streamlines AI inferencing, downstream fine-tuning, and domain-specific evaluation for representation learning models across biomedical modalities. The PyTDC model server (section 3.2) facilitates the use and evaluation of biomedical foundation models for downstream therapeutic tasks and enforces alignment between new method development and therapeutic objectives. We have released support for inference on several state-of-the-art (SoTA) models (Lopez et al., 2018; Theodoris et al., 2023; Cui et al., 2024; Li et al., 2024; ESM Team, 2024) as well as benchmarking software for domain-specific evaluation (Velez-Arce et al., 2024).

PyTDC presents open source model retrieval and deployment software that streamlines AI inferencing and exposes state-of-the-art, research-ready models and training setups for biomedical representation learning models across modalities. In section 3.2 and figure 2, we describe the architectural components of the model server and its integration into the model development lifecycle with the TDC platform. In figure 1, we show the components of PyTDC enabling the full development lifecycle for transfer learning ML methods in therapeutics. Figure 3 also shares how a workflow spanning hundreds-to-thousands of lines of code is simplified to below 30 with PyTDC's model server and data retrieval capabilities.

We present PyTDC, a machine-learning platform providing streamlined training, evaluation, and inference software for single-cell biological foundation models to accelerate research in transfer learning method development in therapeutics (Theodoris, 2024). PyTDC introduces an API-first (Beaulieu et al., 2022) architecture that unifies heterogeneous, continuously updated data sources. The platform introduces a model server, which provides unified access to model weights across distributed repositories and standardized inference endpoints. The model server accelerates research workflows (Kahn et al., 2022) by exposing state-of-the-art, research-ready models and training setups for biomedical representation learning models across modalities. Building upon Therapeutic Data Commons (Huang et al., 2021; 2022; Velez-Arce et al., 2024), we present single-cell therapeutics tasks, datasets, and benchmarks for model development and evaluation.

Below the set of model server models.

C.1. scGPT

scGPT (Cui et al., 2024) is A foundation model for single-cell biology based on a generative pre trained transformer across a repository of over 33 million cells.

Model page. https://huggingface.co/apliko/scGPT

Abstract. Generative pretrained models have achieved remarkable success in various domains such as language and computer vision. Specifically, the combination of large-scale diverse datasets and pretrained transformers has emerged as a promising approach for developing foundation models. Drawing parallels between language and cellular biology (in which texts comprise words; similarly, cells are defined by genes), our study probes the applicability of foundation models to advance cellular biology and genetic research. Using burgeoning single-cell sequencing data, we have constructed a foundation model for single-cell biology, scGPT, based on a generative pretrained transformer across a repository of over 33 million cells. Our findings illustrate that scGPT effectively distills critical biological insights concerning genes and cells. Through further adaptation of transfer learning, scGPT can be optimized to achieve superior performance across diverse downstream applications. This includes tasks such as cell type annotation, multi-batch integration, multi-omic integration, perturbation response prediction and gene network inference.

PyTDC Code. See https://huggingface.co/apliko/scGPT. Illustrated in figure 4.

Figure 4. The below code illustrates a workflow using PyTDC's implementation of (Cui et al., 2024). Such a workflow would take a user hundreds-to-thousands of lines of code to develop. PyTDC allows the user to extract single-cell foundation model embeddings from complex and customized gene expression datasets with less than 30 lines of code. See https://huggingface.co/apliko/scGPT

```python
from tdc_ml.multi_pred.anndata_dataset import DataLoader
from tdc_ml import tdc_hf_interface
from tdc_ml.model_server.tokenizers.scgpt import scGPTTokenizer
import torch

# an example dataset
adata = DataLoader("cellxgene_sample_small",
                   "./data",
                   dataset_names=["cellxgene_sample_small"],
                   no_convert=True).adata

# code for loading the model and performing inference
scgpt = tdc_hf_interface("scGPT")
model = scgpt.load()  # This line can cause segmentation fault on inappropriate setup
tokenizer = scGPTTokenizer()
gene_ids = adata.var["feature_name"].to_numpy(
)  # Convert to numpy array
tokenized_data = tokenizer.tokenize_cell_vectors(
    adata.X.toarray(), gene_ids)
mask = torch.tensor([x != 0 for x in tokenized_data[0][1]],
                    dtype=torch.bool)

# Extract first embedding
first_embed = model(tokenized_data[0][0],
                    tokenized_data[0][1],
                    attention_mask=mask)
```

C.2. Geneformer

Geneformer (Theodoris et al., 2023) is a foundational transformer model pretrained on a large-scale corpus of single cell transcriptomes to enable context-aware predictions in settings with limited data in network biology.

Model page. https://huggingface.co/apliko/Geneformer

Abstract. Mapping gene networks requires large amounts of transcriptomic data to learn the connections between genes, which impedes discoveries in settings with limited data, including rare diseases and diseases affecting clinically inaccessible tissues. Recently, transfer learning has revolutionized fields such as natural language understanding and computer vision by leveraging deep learning models pretrained on large-scale general datasets that can then be fine-tuned towards a vast array of downstream tasks with limited task-specific data. Here, we developed a context-aware, attention-based deep learning model, Geneformer, pretrained on a large-scale corpus of about 30 million single-cell transcriptomes to enable context-specific predictions in settings with limited data in network biology. During pretraining, Geneformer gained a fundamental understanding of network dynamics, encoding network hierarchy in the attention weights of the model in a completely self-supervised manner. Fine-tuning towards a diverse panel of downstream tasks relevant to chromatin and network dynamics using limited task-specific data demonstrated that Geneformer consistently boosted predictive accuracy. Applied to disease modelling with limited patient data, Geneformer identified candidate therapeutic targets for cardiomyopathy. Overall, Geneformer represents a pretrained deep learning model from which fine-tuning towards a broad range of downstream applications can be pursued to accelerate discovery of key network regulators and candidate therapeutic targets.

Code. See https://huggingface.co/apliko/Geneformer. Illustrated in figure 5

Figure 5. The below code illustrates a workflow using PyTDC's implementation of (Theodoris et al., 2023). Such a workflow would take a user hundreds-to-thousands of lines of code to develop. PyTDC allows the user to extract single-cell foundation model embeddings from complex and customized gene expression datasets with less than 30 lines of code. See https://huggingface.co/apliko/Geneformer

```python
from tdc_ml.model_server.tokenizers.geneformer import GeneformerTokenizer
from tdc_ml import tdc_hf_interface
import torch
# Retrieve anndata object. Then, tokenize
tokenizer = GeneformerTokenizer()
x = tokenizer.tokenize_cell_vectors(adata,
                                    ensembl_id="feature_id",
                                    ncounts="n_measured_vars")
cells, _ = x
input_tensor = torch.tensor(cells) # note that you may need to pad or perform other custom
    data processing

# retrieve model
geneformer = tdc_hf_interface("Geneformer")
model = geneformer.load()

# run inference
attention_mask = torch.tensor(
    [[x[0] != 0, x[1] != 0] for x in input_tensor]) # here we assume we used 0/False as a
        special padding token
outputs = model(batch,
                attention_mask=attention_mask,
                output_hidden_states=True)
layer_to_quant = quant_layers(model) + (
    -1
)  # Geneformer's second-to-last layer is most generalized
embs_i = outputs.hidden_states[layer_to_quant]
# there are "cls", "cell", and "gene" embeddings. we will only capture "gene", which is
    cell type specific. for "cell", you'd average out across unmasked gene embeddings per
    cell
embs = embs_i
```

C.3. PINNACLE

PINNACLE (Li et al., 2024), a flexible geometric deep-learning approach that is trained on contextualized protein interaction networks to generate context-PINNACLE protein representations. Leveraging a human multi-organ single-cell transcriptomic atlas, PINNACLE provides 394,760 protein representations split across 156 cell type contexts from 24 tissues and organs.

Model page. https://huggingface.co/apliko/PINNACLE

Abstract. Understanding protein function and developing molecular therapies require deciphering the cell types in which proteins act as well as the interactions between proteins. However, modeling protein interactions across biological contexts remains challenging for existing algorithms. Here we introduce PINNACLE, a geometric deep learning approach that generates context-aware protein representations. Leveraging a multiorgan single-cell atlas, PINNACLE learns on contextualized protein interaction networks to produce 394,760 protein representations from 156 cell type contexts across 24 tissues. PINNACLE's embedding space reflects cellular and tissue organization, enabling zero-shot retrieval of the tissue hierarchy. Pretrained protein representations can be adapted for downstream tasks: enhancing 3D structure-based representations for resolving immuno-oncological protein interactions, and investigating drugs' effects across cell types. PINNACLE outperforms state-of-the-art models in nominating therapeutic targets for rheumatoid arthritis and inflammatory bowel diseases and pinpoints cell type contexts with higher predictive capability than context-free models. PINNACLE's ability to adjust its outputs on the basis of the context in which it operates paves the way for large-scale context-specific predictions in biology.

Code. See https://huggingface.co/apliko/PINNACLE.

Figure 6. The below code illustrates a workflow using PyTDC's implementation of (Li et al., 2024). See https://huggingface.co/apliko/PINNACLE

```
from tdc_ml.resource.pinnacle import PINNACLE
pinnacle = PINNACLE()
embeds = pinnacle.get_embeds() # extract embeddings dataset
```

C.4. scVI

Single-cell variational inference (scVI) (Lopez et al., 2018) is a powerful tool for the probabilistic analysis of single-cell transcriptomics data. It uses deep generative models to address technical noise and batch effects, providing a robust framework for various downstream analysis tasks. To load the pre-trained model, use the Files and Versions tab files.

Code. See https://huggingface.co/apliko/scVI.

Figure 7. The below code illustrates a workflow using PyTDC's implementation of (Lopez et al., 2018). See https://huggingface.co/apliko/scVI

```
from tdc_ml.multi_pred.anndata_dataset import DataLoader
from tdc_ml import tdc_hf_interface

adata = DataLoader("scvi_test_dataset",
                   "./data",
                   dataset_names=["scvi_test_dataset"],
                   no_convert=True).adata

scvi = tdc_hf_interface("scVI")
model = scvi.load()
output = model(adata)
```