

Proposal for an

Open Graph Benchmark

<http://ogb.stanford.edu>

Jure Leskovec, Stanford University

Weihua Hu, Bowen Liu, Marinka Zitnik

Regina Barzilay, Peter Battaglia, Yoshua Bengio, Michael Bronstein,
Stephan Günnemann, Will Hamilton, Tommi Jaakkola,
Stefanie Jegelka, Maximilian Nickel, Chris Re, Le Song, Jian Tang,
Max Welling, Rich Zemel

Graph Representation Learning

- State-of-the-art results in many domains, better applications, new insights, and discoveries!
- Interest in GRL continues to increase
- But...

ML with Graphs

Datasets commonly used today:

- Node classification
 - CORA: 2,708 nodes, 5,429 edges
 - Citeseer: 3,327 nodes, 4,732 edges
- Graph classification
 - MUTAG: 188 molecules (18 nodes each)

ML with Graphs

To properly track progress and identify issues with current approaches it is critical for our community to...

...develop diverse, challenging, and realistic benchmark datasets for machine learning on graphs

Why a New Benchmark?

1) Current focus is on small graphs or small sets of graphs from just a handful of domains:

- Datasets are too small
- Datasets do not contain rich node or edge features
- Hard to reliably and rigorously evaluate algorithms

2) Lack of common benchmark datasets for comparing different methods:

- Every paper design its own, custom train/test split
- Performance across papers is not comparable

3) Dataset splits follow conventional random splits:

- Unrealistic for real-world applications
- Accuracies are over-optimistic under conventional splits

The Open Graph Benchmark

We put together a steering committee

Regina Barzilay, Peter Battaglia,
Yoshua Bengio, Michael Bronstein,
Stephan Günnemann, Will Hamilton,
Tommi Jaakkola, Stefanie Jegelka,
Maximilian Nickel, Chris Re, Le Song,
Jian Tang, Max Welling, Rich Zemel

and we propose the following

The Open Graph Benchmark

OGB is a set of benchmarks for graph ML:

1. Ready-to-use datasets for key tasks on graphs:

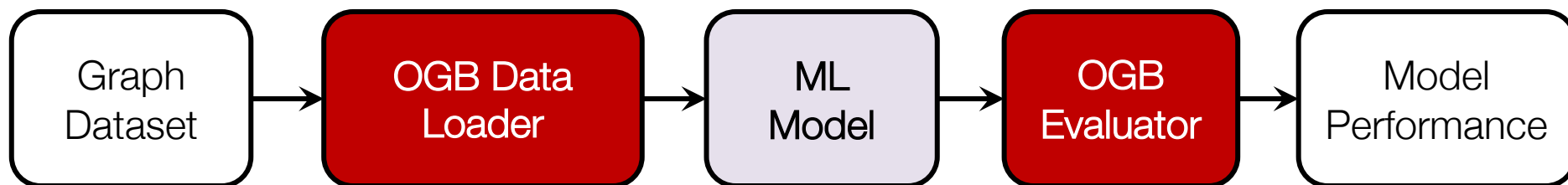
- Node classification, link prediction, graph classification

2. Common codebase to load, construct & represent graphs:

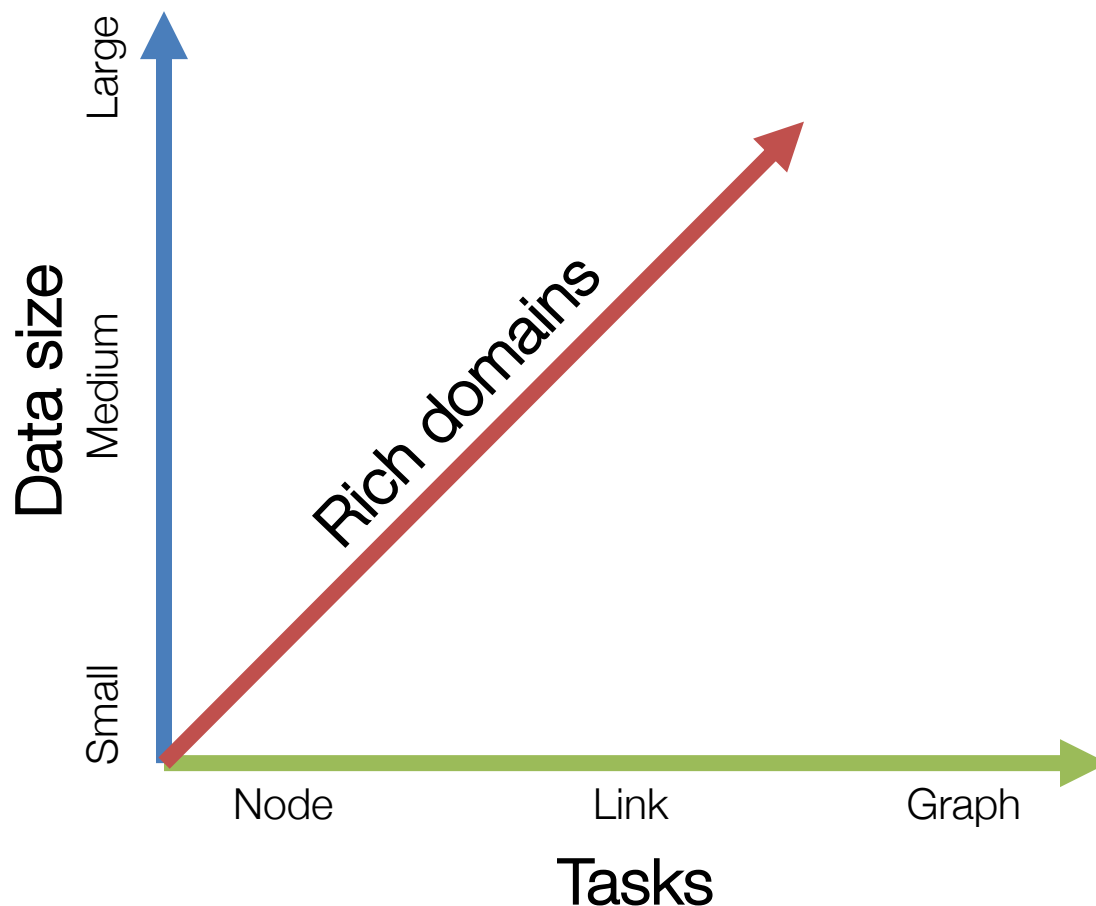
- Popular deep frameworks, e.g., DGL, PyTorch Geometric

3. Common codebase with performance metrics for fast model evaluation and comparison:

- Meaningful train/validation/test splits



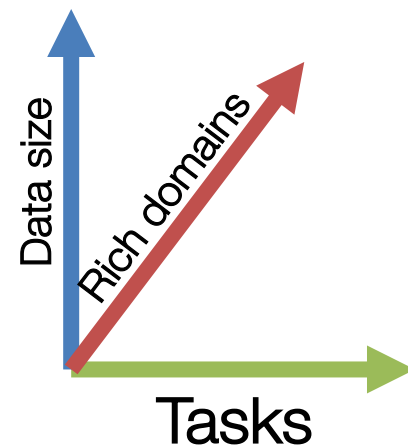
OGB Datasets are Diverse



OGB Datasets are Diverse

Core tasks:

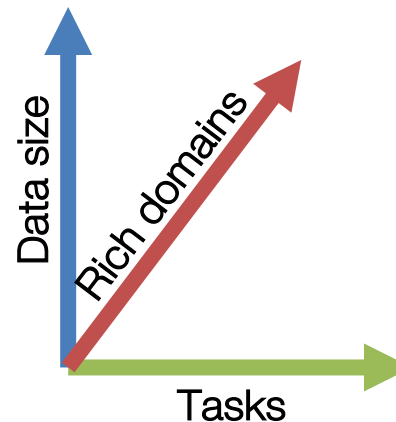
- **Node** property prediction
 - Node label prediction
- **Link** property prediction
 - Link existence prediction
 - Link weight prediction
- **Graph** property prediction
 - Molecule property prediction



OGB Datasets are Diverse

■ Domains:

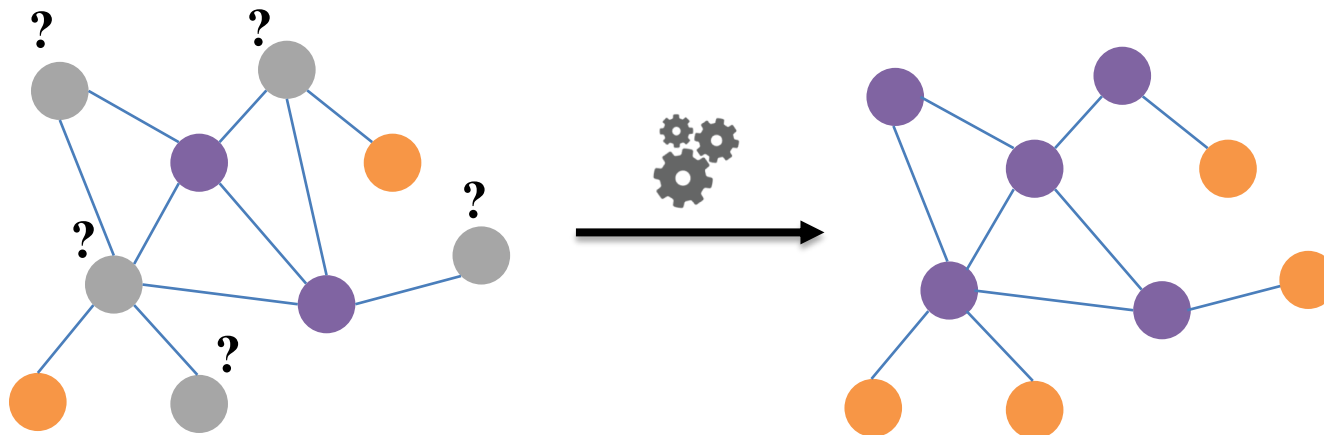
- Natural sciences (chemistry, biology)
- Social and information networks
- Knowledge graphs, code
- **Richness of node/edge/graph features**



■ Dataset size:

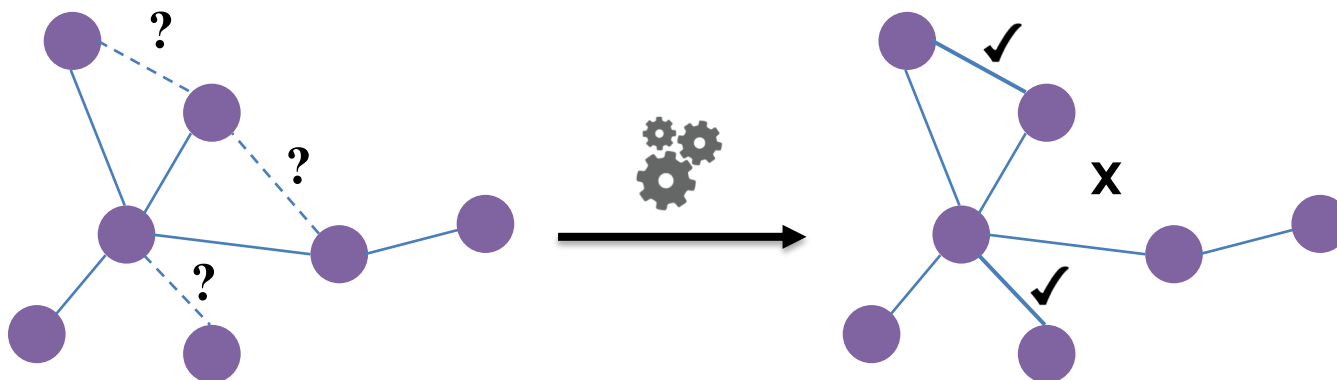
- **Small** rich graphs with 15K to 100K nodes
- **Large** graphs with 200M nodes
- Up to 10M graphs for graph classification

OGB: Node Prediction



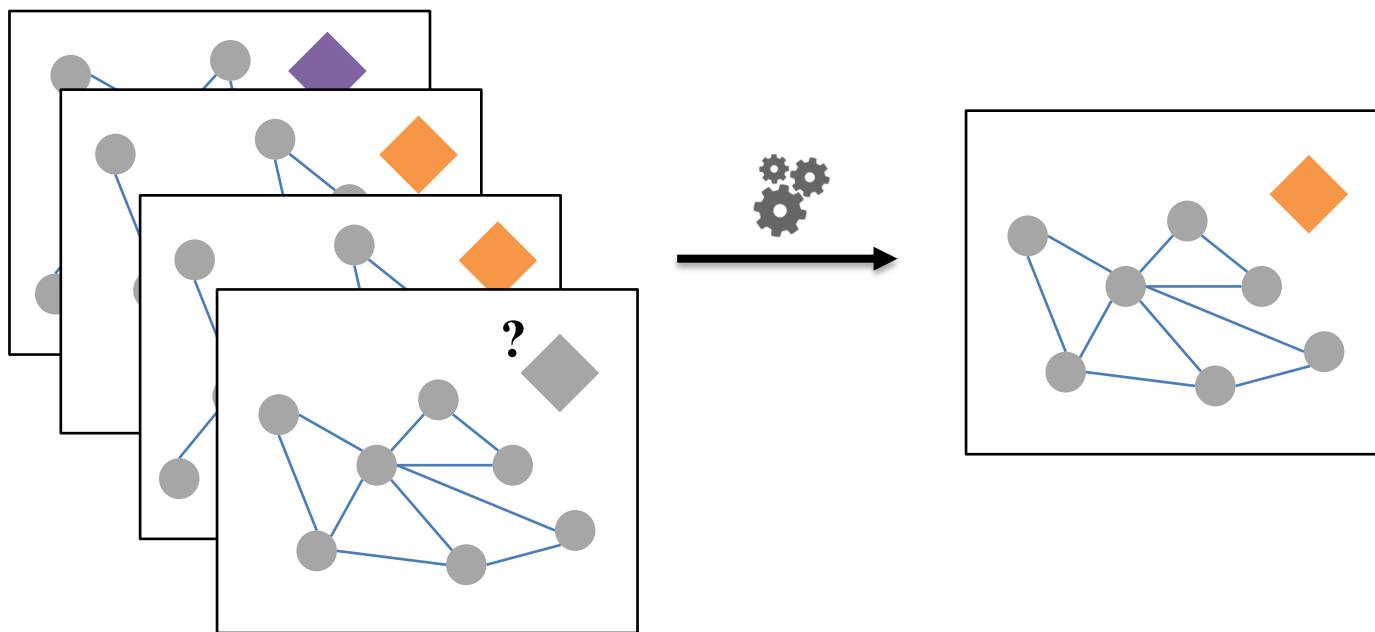
Nodes	Description	ML Task
100K	Protein association network across species	Multi-label binary classification in cellular functions
1M	Wikipedia hyperlinks	Multi-label binary classification in Wikipedia article categories
2M	Amazon co-purchasing network	Multi-label binary classification in Amazon product categories

OGB: Link Prediction



Nodes	Description	Task
15K	Drug-drug interaction network	Signed edge prediction of drug interactions
100K	Human biomedical knowledge graph	Multi-relational edge prediction and multi-hop queries
1M	Protein-protein association network	Weighted multi-relational edge prediction
10M	Amazon user-item review graph	Edge value regression
200M	Microsoft Academic Graph	Edge prediction

OGB: Graph Prediction



Nodes	Description	Task
50K	Molecular graphs	Binary classification/regression on chemical properties of molecules
1M	Abstract syntax trees of code snippets	Multi-label binary classification on semantic properties of code snippets
10M	Protein interaction neighborhoods	Multi-class classification of protein neighborhoods into species

Problem: Random Data Splits

Currently: Prediction accuracies on graph benchmarks range between 75-95%:

- Does stunningly high performance suggest that current models make perfect predictions?

No! Prediction accuracies are often **optimistic under random data splits:**

- Random data splits fail to represent what one would expect in real-world settings
- The use of models can be limited in practice

We need meaningful data splits!

Meaningful Data Splits

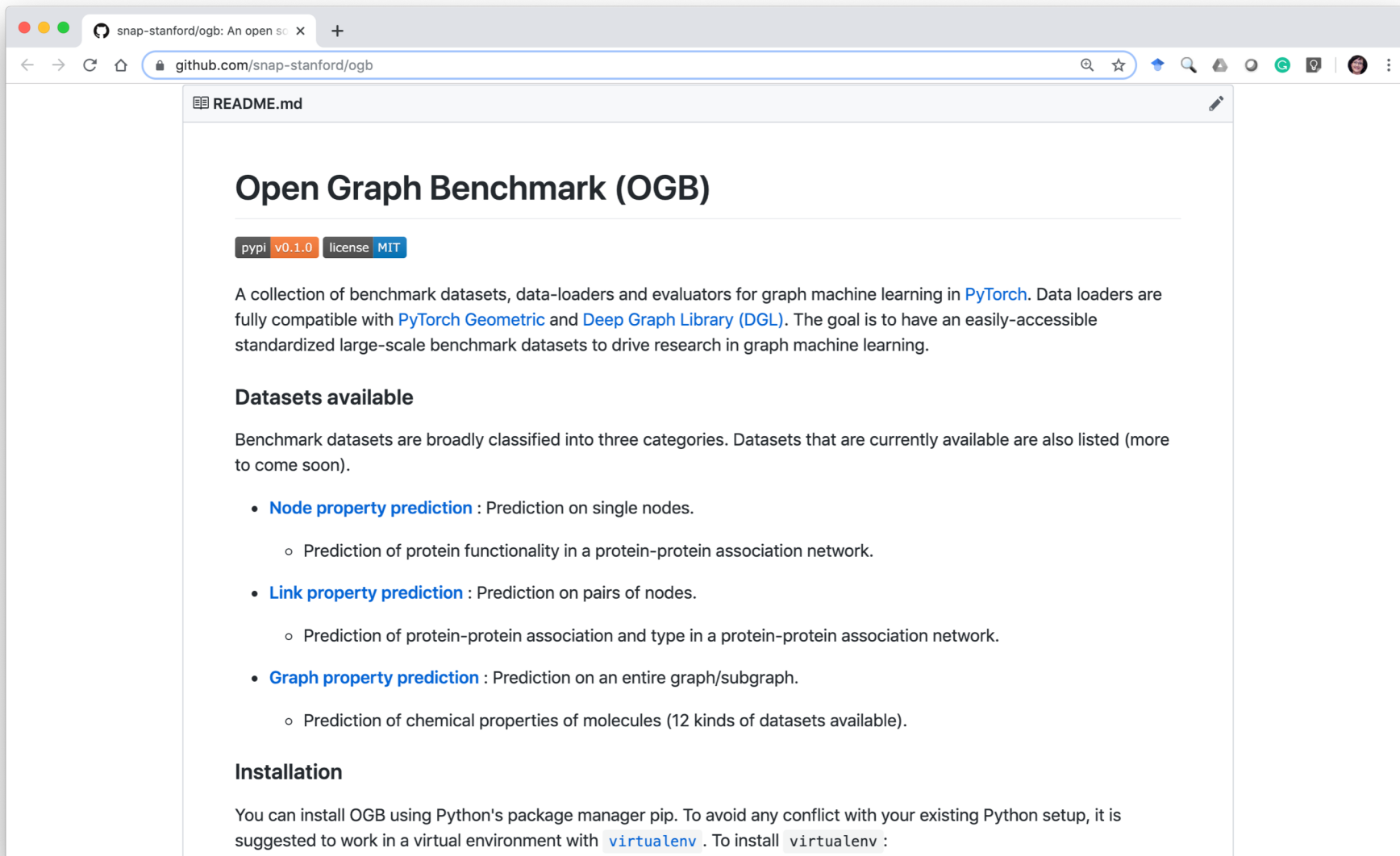
Three examples:

- **Scaffold split:** For molecular graph datasets, OGB:
 - Clusters molecules by scaffold (molecular graph substructure)
 - Combines clusters by placing most common scaffolds in train set
 - Gives validation/test sets with structurally different molecules
- **Species split:** For protein interaction datasets, OGB:
 - Uses protein graphs from model species (weed, worm, E. coli, fly, mouse, yeast, zebrafish) as train/validation sets
 - Uses protein graphs from humans as test set
- **Repository split:** For code snippet datasets, OGB:
 - Groups snippets by code repositories they appear in
 - Places snippets from some code repositories in train set
 - Validation/test sets with snippets using different coding style

Open Graph Benchmark

<https://ogb.stanford.edu>
ogb@cs.stanford.edu

Pip-Installable OGB Package



The screenshot shows a web browser window displaying the GitHub README for the 'snap-stanford/ogb' repository. The browser's address bar shows 'github.com/snap-stanford/ogb'. The README title is 'Open Graph Benchmark (OGB)'. Below the title, there are badges for 'pypi v0.1.0' and 'license MIT'. The main text describes the project as a collection of benchmark datasets, data-loaders, and evaluators for graph machine learning in PyTorch, compatible with PyTorch Geometric and Deep Graph Library (DGL). It states the goal is to have easily-accessible standardized large-scale benchmark datasets. The 'Datasets available' section lists three categories: Node property prediction, Link property prediction, and Graph property prediction, each with specific examples. The 'Installation' section provides instructions on how to install the package using pip and suggests using a virtual environment with virtualenv.

Open Graph Benchmark (OGB)

`pypi v0.1.0` `license MIT`

A collection of benchmark datasets, data-loaders and evaluators for graph machine learning in [PyTorch](#). Data loaders are fully compatible with [PyTorch Geometric](#) and [Deep Graph Library \(DGL\)](#). The goal is to have an easily-accessible standardized large-scale benchmark datasets to drive research in graph machine learning.

Datasets available

Benchmark datasets are broadly classified into three categories. Datasets that are currently available are also listed (more to come soon).

- [Node property prediction](#) : Prediction on single nodes.
 - Prediction of protein functionality in a protein-protein association network.
- [Link property prediction](#) : Prediction on pairs of nodes.
 - Prediction of protein-protein association and type in a protein-protein association network.
- [Graph property prediction](#) : Prediction on an entire graph/subgraph.
 - Prediction of chemical properties of molecules (12 kinds of datasets available).

Installation

You can install OGB using Python's package manager pip. To avoid any conflict with your existing Python setup, it is suggested to work in a virtual environment with [virtualenv](#). To install `virtualenv` :

OGB Data Loaders, Evaluators

■ Easy-to-use dataset loaders in PyTorch:

```
from ogb.graphproppred.dataset_pyg import PygGraphPropPredDataset
from torch_geometric.data import DataLoader

dataset = PygGraphPropPredDataset(name = "ogbg-mol-tox21")
splitted_idx = dataset.get_split_idx()

train_loader = DataLoader(dataset[splitted_idx["train"]], batch_size=32, shuffle=True)
valid_loader = DataLoader(dataset[splitted_idx["valid"]], batch_size=32, shuffle=False)
test_loader = DataLoader(dataset[splitted_idx["test"]], batch_size=32, shuffle=False)
```

```
from ogb.graphproppred import Evaluator

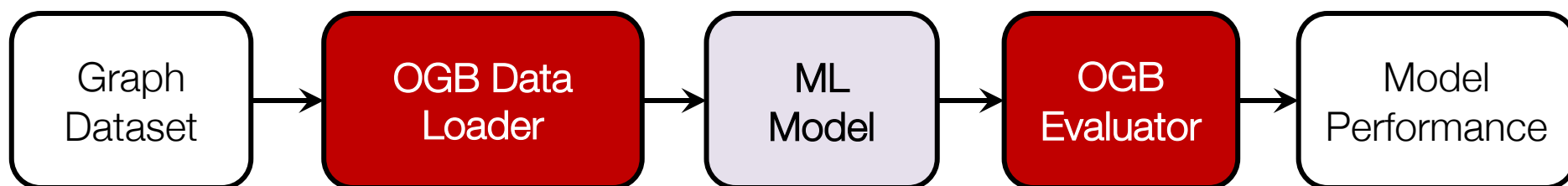
evaluator = Evaluator(name = "ogbg-mol-tox21")
# We can learn the input and output format specification of the evaluator as follows.
# print(evaluator.expected_input_format)
# print(evaluator.expected_output_format)
input_dict = {"y_true": y_true, "y_pred": y_pred}
result_dict = evaluator.eval(input_dict) # E.g., {"ap": 0.3421, "rocauc": 0.7321}
```

Open Graph Benchmark

Resource for a range of graph ML problems

We envision OGB to lead to many applications:

- Teaching resource,
- Database of benchmark datasets,
- Realistic environment for new tasks
- Opportunities for new research in graph representation learning and beyond



Discussion Points

More to come at <http://ogb.stanford.edu>

- More datasets
- More tasks
- Leaderboards

How to get involved:

- Provide feedback
- Propose datasets
- Benchmark models

Contact us at ogb@cs.stanford.edu

Open Graph Benchmark

<https://ogb.stanford.edu>
ogb@cs.stanford.edu

Core development team

Bowen Liu, Weihua Hu, Marinka Zitnik, Jure Leskovec

Steering committee

Regina Barzilay, Peter Battaglia, Yoshua Bengio, Michael Bronstein, Stephan Günnemann, Will Hamilton, Tommi Jaakkola, Stefanie Jegelka, Maximilian Nickel, Chris Re, Le Song, Jian Tang, Max Welling, Rich Zemel