

리얼딥바이오 파트1-2

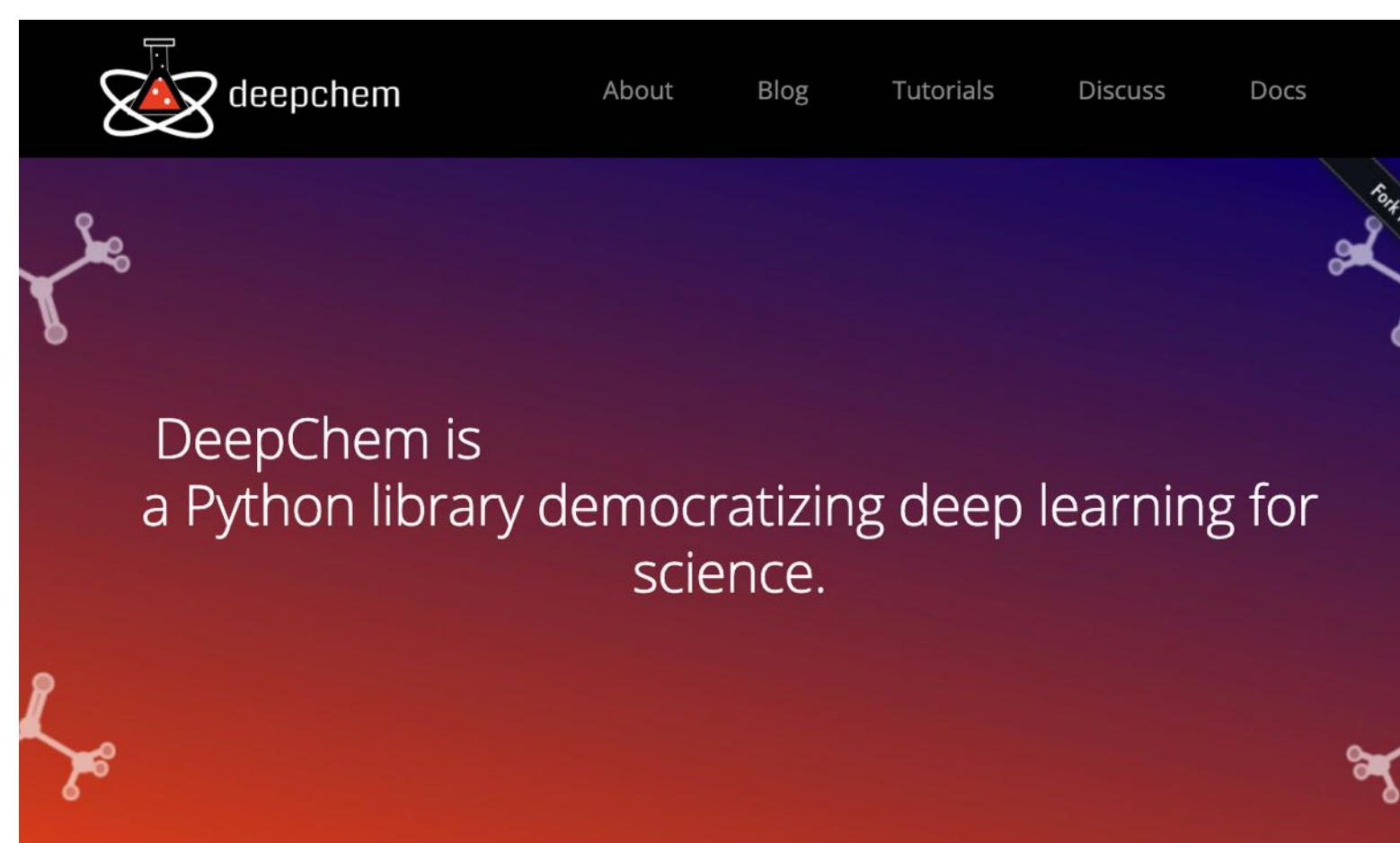
# Chapter 3. Machine Learning with DeepChem

고준수

2019-06-21

# What is DeepChem?

*“a python library that provides a high quality open-source toolchain for deep-learning in drug discovery, materials science, quantum chemistry, and biology.”*



## Get Started.

Select your preferences, then run the DeepChem install command.

Anaconda is our recommended package manager

OS	<input type="radio"/> Linux	<input checked="" type="radio"/> OSX
Python	<input type="radio"/> 2.7	<input checked="" type="radio"/> 3.5
GPU Enabled	<input type="radio"/> YES	<input checked="" type="radio"/> NO
Package Manager	<input type="radio"/> conda	<input checked="" type="radio"/> docker

Run this command:

Follow instructions at our github:  
<https://github.com/deepchem/deepchem>

<https://deepchem.io/>

- This chapter provides a brief introduction to machine learning with DeepChem, a library built on top of the TensorFlow platform to facilitate the use of deep learning in the life sciences.
- DeepChem provides a large collection of models, algorithms, and datasets that are suited to applications in the life sciences.
- In the remainder of this book, we will use DeepChem to perform our case studies.

# Contents

- DeepChem Datasets
- Training a Model to Predict Toxicity of Molecules
- Case Study: Training an MNIST Model
  - The MNIST Digit Recognition Dataset
  - A Convolutional Architecture for MNIST
- SoftMax and SoftMaxCrossEntropy
- Conclusion

# 환경 설정

```
$ sudo apt-get install python-rdkit librdkit-dev
```

```
$ virtualenv --python=python2.7 realdeepbio
```

```
$ cd realdeepbio/lib/python2.7/site-packages/
```

```
$ cp -r /usr/lib/python2.7/dist-packages/rdkit .
```

```
$ cd -
```

```
$ source realdeepbio/bin/activate
```

```
(realdeepbio)$ pip install numpy
```

```
(realdeepbio)$ pip install joblib
```

```
(realdeepbio)$ pip install pandas
```

```
(realdeepbio)$ pip install sklearn
```

```
(realdeepbio)$ pip install tensorflow
```

```
(realdeepbio)$ pip install pillow
```

```
(realdeepbio)$ pip install deepchem
```

```
(realdeepbio)$ python
```

```
Python 2.7.12 (default, Nov 12 2018, 14:36:49)
```

```
[GCC 5.4.0 20160609] on linux2
```

```
Type "help", "copyright", "credits" or "license" for more information.
```

```
>>> import deepchem
```

```
>>>
```

- Unbutu 16.04
- Python 2.7

# DeepChem Datasets

- **Dataset** : DeepChem에서 사용하는 data wrapping 오브젝트
  - x : input vectors
  - y : output vectors
  - other information such as a description of what each sample represents
- Subclass : 다양한 형식의 데이터 저장 형식을 가지고 있는 오브젝트
  - **NumpyDataset** : convenient wrapper for NumPy arrays

# Example) NumpyDataset

```
>>> import deepchem as dc
>>> import numpy as np

>>> x = np.random.random((4, 5))
>>> y = np.random.random((4, 1))
>>> dataset = dc.data.NumpyDataset(x, y)

>>> print(dataset.X)
[[0.65299758 0.30423263 0.02754099 0.97309579 0.153431   ]
 [0.88351812 0.53595171 0.04685409 0.54723345 0.8166021   ]
 [0.70563828 0.39927295 0.02356079 0.43582273 0.95855071]
 [0.7639138  0.99966694 0.03611058 0.43419459 0.20166431]]

>>> print(dataset.y)
[[0.24408776]
 [0.14880403]
 [0.87203699]
 [0.5207757  ]]
```

- NumpyDataset 생성 예시
  - **X** : 5개의 feature를 가지는 4개의 샘플
    - (**대문자**)
  - y : 4개 샘플의 결과 값
    - (**소문자**).

# Training a Model to Predict Toxicity of Molecules

- 목적
  - DeepChem을 이용하여 분자의 독성을 예측하는 모델을 train 하는 방법
  - 기계학습 문제를 해결하기 위해서 DeepChem을 사용하는 방법



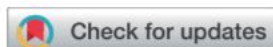
# MoleculeNet

Chemical  
Science



EDGE ARTICLE

View Article Online  
View Journal | View Issue



Cite this: *Chem. Sci.*, 2018, 9, 513

## MoleculeNet: a benchmark for molecular machine learning†

Zhenqin Wu,<sup>‡a</sup> Bharath Ramsundar,<sup>‡b</sup> Evan N. Feinberg,<sup>§c</sup> Joseph Gomes,<sup>‡a</sup> Caleb Geniesse,<sup>c</sup> Aneesh S. Pappu,<sup>b</sup> Karl Leswing<sup>d</sup> and Vijay Pande<sup>\*a</sup>

Molecular machine learning has been maturing rapidly over the last few years. Improved methods and the presence of larger datasets have enabled machine learning algorithms to make increasingly accurate predictions about molecular properties. However, algorithmic progress has been limited due to the lack of a standard benchmark to compare the efficacy of proposed methods; most new algorithms are benchmarked on different datasets making it challenging to gauge the quality of proposed methods. This work introduces MoleculeNet, a large scale benchmark for molecular machine learning. MoleculeNet curates multiple public datasets, establishes metrics for evaluation, and offers high quality open-source implementations of multiple previously proposed molecular featurization and learning algorithms (released as part of the DeepChem open source library). MoleculeNet benchmarks demonstrate that learnable representations are powerful tools for molecular machine learning and broadly offer the best performance. However, this result comes with caveats. Learnable representations still struggle to deal with complex tasks under data scarcity and highly imbalanced classification. For quantum mechanical and biophysical datasets, the use of physics-aware featurizations can be more important than choice of particular learning algorithm.

Received 15th June 2017  
Accepted 30th October 2017

DOI: 10.1039/c7sc02664a

rsc.li/chemical-science

## 1 Introduction

Overlap between chemistry and statistical learning has had a long history. The field of cheminformatics has been utilizing machine learning methods in chemical modeling (*e.g.* quantitative structure activity relationships, QSAR) for decades.<sup>1–6</sup> In the recent 10 years, with the advent of sophisticated deep learning methods,<sup>7,8</sup> machine learning has gathered increasing amounts of attention from the scientific community. Data-driven analysis has become a routine step in many chemical and biological applications, including virtual screening,<sup>9–12</sup> chemical property prediction,<sup>13–16</sup> and quantum chemistry calculations.<sup>17–20</sup>

In many such applications, machine learning has shown strong potential to compete with or even outperform conventional *ab initio* computations.<sup>16,18</sup> It follows that introduction of novel machine learning methods has the potential to reshape research on properties of molecules. However, this potential

has been limited by the lack of a standard evaluation platform for proposed machine learning algorithms. Algorithmic papers often benchmark proposed methods on disjoint dataset collections, making it a challenge to gauge whether a proposed technique does in fact improve performance.

Data for molecule-based machine learning tasks are highly heterogeneous and expensive to gather. Obtaining precise and accurate results for chemical properties typically requires specialized instruments as well as expert supervision (contrast with computer speech and vision, where lightly trained workers can annotate data suitable for machine learning systems). As a result, molecular datasets are usually much smaller than those available for other machine learning tasks. Furthermore, the breadth of chemical research means our interests with respect to a molecule may range from quantum characteristics to measured impacts on the human body. Molecular machine learning methods have to be capable of learning to predict this very broad range of properties. Complicating this challenge, input molecules can have arbitrary size and components, highly variable connectivity and many three dimensional conformers (three dimensional molecular shapes). To transform molecules into a form suitable for conventional machine learning algorithms (that usually accept fixed length input), we have to extract useful and related information from a molecule into a fixed dimensional representation (a process called featurization).<sup>21–23</sup>

To put it simply, building machine learning models on molecules requires overcoming several key issues: limited

## MoleculeNet

### A Benchmark for Molecular Machine Learning

Category	Dataset	Data Type	Task Type	# Tasks	# Compounds	Rec - Split <sup>a</sup>	Rec - Metric <sup>b</sup>
Quantum Mechanics	QM7	SMILES, 3D coordinates	Regression	1	7160	Stratified	MAE
	QM7b	3D coordinates	Regression	14	7210	Random	MAE
	QM8	SMILES, 3D coordinates	Regression	12	21786	Random	MAE
	QM9	SMILES, 3D coordinates	Regression	12	133885	Random	MAE
Physical Chemistry	ESOL	SMILES	Regression	1	1128	Random	RMSE
	FreeSolv	SMILES	Regression	1	642	Random	RMSE
	Lipophilicity	SMILES	Regression	1	4200	Random	RMSE
Biophysics	PCBA	SMILES	Classification	128	437929	Random	PRC-AUC
	MUV	SMILES	Classification	17	93087	Random	PRC-AUC
	HIV	SMILES	Classification	1	41127	Scaffold	ROC-AUC
	PDBbind	SMILES, 3D coordinates	Regression	1	11908	Time	RMSE
Physiology	BACE	SMILES	Classification	1	1513	Scaffold	ROC-AUC
	BBBP	SMILES	Classification	1	2039	Scaffold	ROC-AUC
	Tox21	SMILES	Classification	12	7831	Random	ROC-AUC
	ToxCast	SMILES	Classification	617	8575	Random	ROC-AUC
	SIDER	SMILES	Classification	27	1427	Random	ROC-AUC
	ClinTox	SMILES	Classification	2	1478	Random	ROC-AUC

<http://moleculenet.ai/>

Quantum Mechanics

- **QM7/QM7b**: Electronic properties(atomization energy, HOMO/LUMO, etc.) determined using *ab-initio* density functional theory(DFT).

Regression  
3D Coordinates

- **QM8**: Electronic spectra and excited state energy of small molecules calculated by multiple quantum mechanic methods.

Regression  
3D Coordinates

- **QM9**: Geometric, energetic, electronic and thermodynamic properties of DFT-modelled small molecules.

Regression  
3D Coordinates

Physical Chemistry

- **ESOL**: Water solubility data(log solubility in mols per litre) for common organic small molecules.

Regression

- **FreeSolv**: Experimental and calculated hydration free energy of small molecules in water.

Regression

- **Lipophilicity**: Experimental results of octanol/water distribution coefficient(logD at pH 7.4).

Regression

Biophysics

- **PCBA**: Selected from PubChem BioAssay, consisting of measured biological activities of small molecules generated by high-throughput screening.

Classification

- **MUV**: Subset of PubChem BioAssay by applying a refined nearest neighbor analysis, designed for validation of virtual screening techniques.

Classification

- **HIV**: Experimentally measured abilities to inhibit HIV replication.

Classification

- **PDBbind**: Binding affinities for bio-molecular complexes, both structures of proteins and ligands are provided.

Regression  
3D Coordinates

- **BACE**: Quantitative (IC50) and qualitative (binary label) binding results for a set of inhibitors of human  $\beta$ -secretase 1(BACE-1).

Regression  
Classification

Physiology

- **BBBP**: Binary labels of blood-brain barrier penetration(permeability).

Classification

- **Tox21**: Qualitative toxicity measurements on 12 biological targets, including nuclear receptors and stress response pathways.

Classification

- **ToxCast**: Toxicology data for a large library of compounds based on *in vitro* high-throughput screening, including experiments on over 600 tasks.

Classification

- **SIDER**: Database of marketed drugs and adverse drug reactions (ADR), grouped into 27 system organ classes.

Classification

- **ClinTox**: Qualitative data of drugs approved by the FDA and those that have failed clinical trials for toxicity reasons.

Classification

<sup>a</sup>Department of Chemistry, Stanford University, Stanford, CA 94305, USA. E-mail: pande@stanford.edu

<sup>b</sup>Department of Computer Science, Stanford University, Stanford, CA 94305, USA

<sup>c</sup>Program in Biophysics, Stanford School of Medicine, Stanford, CA 94305, USA

<sup>d</sup>Schrodinger Inc., USA

† Electronic supplementary information (ESI) available. See DOI: 10.1039/c7sc02664a

‡ Joint first authorship.

§ Joint second authorship.



# Tox21 Data Challenge 2014

**National Toxicology Program**  
U.S. Department of Health and Human Services

[Calendar & Events](#) | 
 [News & Media](#) | 
 [Get Involved](#) | 
 [Support](#)

[Testing Information](#) ▾ | 
 [Study Results & Research Projects](#) ▾ | 
 [Public Health](#) ▾ | 
 [About NTP](#) ▾

- ▶ Testing Status of Substances at NTP
- ▶ Study Types
- ▶ Nominations to the Testing Program

## Toxicology in the 21st Century (Tox21)

Thousands of chemical substances exist in the world, but only a small fraction of these have been adequately assessed for their potential toxicity to humans. The Toxicology in the 21st Century program, or Tox21, is a unique collaboration between several federal agencies to develop new ways to rapidly test whether substances adversely affect human health. Substances assayed in Tox21 include a diverse range of products such as: commercial chemicals, pesticides, food additives/contaminants, and medical compounds.

The following four government agencies bring their unique expertise, resources, and tools to the Tox21 collaboration:

- **National Institute of Environmental Health Sciences (NIEHS) / National Toxicology Program (NTP),** National Institutes of Health (NIH), U.S. Department of Health and Human Services
- **National Center for Advancing Translational Sciences (NCATS),** National Institutes of Health (NIH)
- **U.S. Food and Drug Administration (FDA),** U.S. Department of Health and Human Services
- **National Center for Computational Toxicology,** Office of Research and Development, U.S. Environmental Protection Agency (EPA)

**Robotic arm used for Tox21 testing**

**SHARE THIS:**

https://ntp.niehs.nih.gov/go/tox21 ↗

- ▾ On This Page
  - [Toxicology in the 21st Century \(Tox21\)](#)
  - [Goals of the Tox21 Program](#)
  - [Research Phases](#)
  - [Contact](#)
- ▾ Related Links
  - [Biomolecular Screening Branch \(BSB\)](#) ↗
  - [Tox21: Chemical Testing in the 21st Century](#) ↗
  - [United States Federal Government Collaboration Toxicology in the 21st Century \(Tox21\): Testing Thousands of Environmental Chemicals Using Non-Animal Methods](#) ↗

[BACK TO TOP](#)

 U.S. Department of Health & Human Services

 National Institutes of Health



National Center  
for Advancing  
Translational Sciences

Tox21 Data Challenge 2014

Login

Home

» About

Registration

Data/Resources

Submissions

Leaderboard

Contact Us

Survey 





About the Challenge

What is Tox 21? 



Most people are exposed to many

Challenge Overview

The goal of the challenge is to “crowdsource” data analysis by independent researchers to reveal how well they can predict compounds’ interference in biochemical pathways using only chemical structure data. The computational models produced from the Challenge could become decision-making tools for government agencies in determining which environmental chemicals and drugs are of the greatest potential concern to human health.


























The Challenge

Use data generated from nuclear receptor signaling and stress pathway assays run against Tox21’s 10,000-compound library (Tox21 10K) to build models and look for structure-activity relationships.

Participants have the opportunity to join 15 distinct subchallenges:

- Subchallenges 1–12 use data from a single assay.
- Subchallenges 13 and 14 capture all assays of a specific type (nuclear receptor signaling (NR) or stress response (SR)).
- Subchallenge 15 captures all assays of both types.

Participants can choose to join one or all of the subchallenges. There will be one winner for each subchallenge.

Training Datasets 		
<p>The complete training dataset is available <a href="#">here</a>. For individual datasets, please use the links below. In the datasets, "1" means active, "0" means inactive.</p>		
Nuclear Receptor Signaling Panel		
Assay	SDF	SMILES
AR		
AhR		
AR-LBD		
ER		
ER-LBD		
aromatase		
PPAR-gamma		
Stress Response Panel		
Assay	SDF	SMILES
ARE		
ATAD5		
HSE		
MMP		
p53		



# Toxicity Prediction : Tox21

```
>>> import deepchem as dc
>>> import numpy as np

>>> tox21_tasks, tox21_datasets, transformers = dc.molnet.load_tox21()
Loading raw samples now.
shard_size: 8192
About to start loading CSV from /tmp/tox21.csv.gz
Loading shard 1 of size 8192.
Featurizing sample 0
Featurizing sample 1000
Featurizing sample 2000
Featurizing sample 3000
Featurizing sample 4000
Featurizing sample 5000
Featurizing sample 6000
Featurizing sample 7000
TIMING: featurizing shard 0 took 11.062 s
TIMING: dataset construction took 11.450 s
Loading dataset from disk.
TIMING: dataset construction took 0.435 s
Loading dataset from disk.
TIMING: dataset construction took 0.414 s
Loading dataset from disk.
TIMING: dataset construction took 0.216 s
Loading dataset from disk.
TIMING: dataset construction took 0.211 s
>>>
```

- dc.molnet
  - MoleculeNet 데이터를 다루는 모듈
- Tox21
- *featurization*
  - how a dataset containing information about molecules is transformed into matrices and vectors for use in machine learning analyses.
-

# Tox21 : dataset

```
>>> tox21_tasks
[u'NR-AR', u'NR-AR-LBD', u'NR-AhR', u'NR-Aromatase', u'NR-ER', u'NR-ER-
LBD', u'NR-PPAR-gamma', u'SR-ARE', u'SR-ATAD5', u'SR-HSE', u'SR-MMP',
u'SR-p53']
>>> len(tox21_tasks)
12

>>> tox21_datasets
(<deepchem.data.datasets.DiskDataset object at 0x7f5ec3951f10>,
<deepchem.data.datasets.DiskDataset object at 0x7f5ec3977f10>,
<deepchem.data.datasets.DiskDataset object at 0x7f5ec393c510>)

>>> len(tox21_datasets)
3
>>> train_dataset, valid_dataset, test_dataset = tox21_datasets
>>> train_dataset.X.shape
(6264, 1024)
>>> valid_dataset.X.shape
(783, 1024)
>>> test_dataset.X.shape
(784, 1024)

>>> np.shape(train_dataset.y)
(6264, 12)
>>> print(train_dataset.y[:3])
[[0. 0. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
```

- tox21\_tasks
  - 12개의 task가 있으며, 각각 다른 생물학적 실험의 결과임.
  - 각각 독성과 관련있는 개별 효소 반응과 연관성이 있다고 생각됨.
- tox21\_datasets
  - train, valid, test 가 모두 나누져 있음.
- transformers



# Tox21 : Missing & $w$ (weights)

- Tox21 실험데이터는 실제로는 모든 molecule이 모든 실험을 수행하지 않았음.
  - Label 중 일부는 의미없는 결과를 담고 있음.
  - Training과 Testing 과정에서 이런 의미없는 정보를 담고 있는 요소들을 제거해야할 필요가 있음.
- 데이터의  $w$  값을 이용해서 확인. (weights)
  - Whenever we compute the loss function for a model, we multiply by  $w$  before summing over tasks and samples.

```
>>> train_dataset.w.shape
(6264, 12)
>>> np.count_nonzero(train_dataset.w)
62166
>>> np.count_nonzero(train_dataset.w == 0)
13002
```

$$6,264 \times 12 = 75,168$$

# transformer

- DeepChem은 data loading 시에 데이터 보정에 사용한 transformer 등을 같이 제공한다.
- BalancingTransformer
  - Unbalanced 데이터를 수정하기 위해서 사용됨.
    - Tox21의 경우 90% 이상의 데이터가 0으로 분류됨.
    - 무조건 0으로 예측하면 90%의 정확도를 가진다는 의미
  - 가장 좋은 방법은 data의 weights matrix를 보정하는 것
  - BalancingTransformer 는 각 class의 weight 총합이 동일하도록 각 data point 의 weight를 보정함.

```
>>> transformers
[<deepchem.trans.transformers.BalancingTransformer object
at 0x7f5ef44cb3d0>]
>>>
```

1개의 transformer 가 사용된 것을 확인할 수 있음.

# dc.models

- dc.models 의 하위 모듈은 다양한 모델을 담고 있음.
  - 모든 dc.models.Model 을 상속하고 있음.
- dc.models.MultitaskClassifier
  - This model builds a fully connected network (an MLP) that maps input features to multiple output predictions.
  - This makes it useful for multitask problems, where there are multiple labels for every sample.

```
>>> model =  
dc.models.MultitaskClassifier(n_tasks=12,n_features=  
1024, layer_sizes=[1000])
```

- n\_tasks
  - Number of tasks
- n\_features
  - Number of input features
- layer\_sizes
  - a list that sets the number of fully connected hidden layers in the network, and the width of each one.
  - a single hidden layer of width 1,000.



# train

- `fit()` : Fits the model to the data contained in a Dataset object.
  - `epoch`
    - refers to one complete pass through all the samples in a dataset
  - `batch`
    - the training set into batches and take one step of gradient descent for each batch

```
>>> model.fit(train_dataset, nb_epoch=10)
WARNING: Logging before flag parsing goes to stderr.
.....
803.1958665151444
>>>
```

- `nb_epoch=10`
  - 10 epochs of gradient descent training
  -

# Evaluate the performance of trained model

- Model이 얼마자 잘 작동하는지를 알아보자
- For the Tox21 datasets,
  - the ROC AUC score is a useful metric, so let's do our analysis using it.
  - there are multiple Tox21 tasks.
  - A good tactic is to compute the **mean ROC AUC score** across all tasks.

```
>>> metric = dc.metrics.Metric(dc.metrics.roc_auc_score, np.mean)
>>> metric
<deepchem.metrics.Metric object at 0x7f5ec3977350>
>>> train_scores = model.evaluate(train_dataset, [metric], transformers)
>>> test_scores = model.evaluate(test_dataset, [metric], transformers)

>>> print(train_scores)
{'mean-roc_auc_score': 0.9665283297494872}
>>> print(test_scores)
{'mean-roc_auc_score': 0.7923191024381578}
```

- np.mean
  - the mean of the ROC AUC scores
- our score on the training set (0.96) is much better than our score on the test set (0.79).
  - This shows the model has been **overfit**.

# Case Study : Training an MNIST Model

- We used a **premade model class**, `dc.models.MultitaskClassifier`.
- **Creating a new deep learning architecture** instead of using a preconfigured one.
- how to train a convolutional neural network on the MNIST digit recognition dataset.
- We will **specify the full deep learning architecture ourselves**.
- To do so, we will introduce the `dc.models.TensorGraph` class.



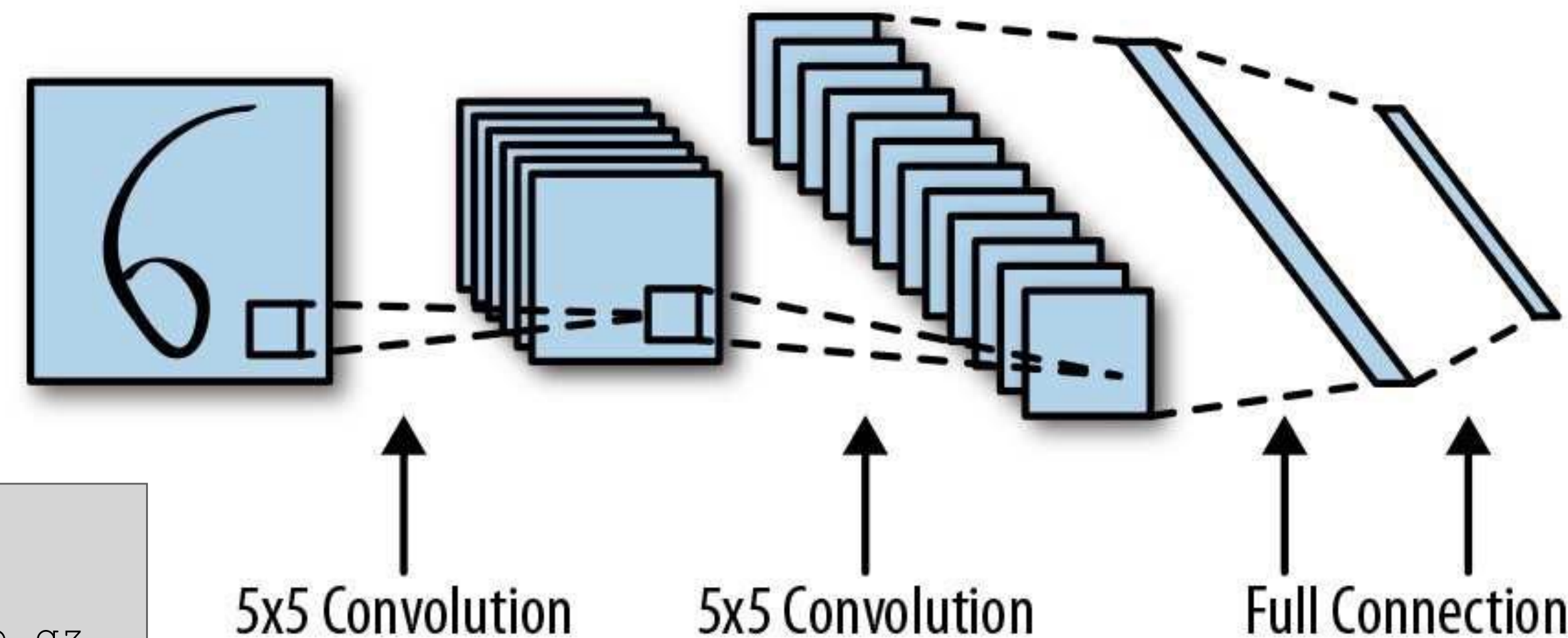
# The MNIST Digit Recognition Dataset



- a machine learning model that can learn to classify handwritten digits correctly.
- The challenge is to classify digits from 0 to 9 given  $28 \times 28$ -pixel black and white images.
- The dataset contains 60,000 training examples and a test set of 10,000 examples.

# A Convolutional Architecture for MNIST

- DeepChem uses the TensorGraph class to construct nonstandard deep learning architectures.
- They are followed by **two fully connected layers** to predict the digit from those local features.



```
$ mkdir MNIST_data
$ cd MNIST_data
$ wget http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz
$ wget http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz
$ wget http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz
$ wget http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz
$ cd ..
```

# 데이터 형식 변환

- to process this raw data into a format suitable for analysis by DeepChem
- `deepchem.models.tensorgraph.layers`
  - contains a collection of "**layers**."
  - These layers serve as **building blocks of deep architectures** and can be composed to build new deep learning architectures.
- We will demonstrate how layer objects are used shortly. Next, we construct NumpyDataset objects that wrap the MNIST training and test datasets:

```
import deepchem as dc
import tensorflow as tf
import deepchem.models.tensorgraph.layers as layers

from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)

train_dataset = dc.data.NumpyDataset(mnist.train.images,mnist.train.labels)
test_dataset = dc.data.NumpyDataset(mnist.test.images,mnist.test.labels)
```

- `input_data()` function
  - takes care of **separating out a proper test** dataset for our use.



# To build new models

- each layer takes input from previous layers and computes an output that can be passed to subsequent layers

```
model = dc.models.TensorGraph(model_dir='mnist')

feature = layers.Feature(shape=(None, 784))
label = layers.Label(shape=(None, 10))

make_image = layers.Reshape(shape=(None, 28, 28), in_layers=feature)

conv2d_1 = layers.Conv2D(num_outputs=32, activation_fn=tf.nn.relu,
in_layers=make_image)
conv2d_2 = layers.Conv2D(num_outputs=64, activation_fn=tf.nn.relu,
in_layers=conv2d_1)

flatten = layers.Flatten(in_layers=conv2d_2)
dense1 = layers.Dense(out_channels=1024, activation_fn=tf.nn.relu, in_layers=flatten)
dense2 = layers.Dense(out_channels=10, activation_fn=None, in_layers=dense1)
smce = layers.SoftMaxCrossEntropy(in_layers=[label, dense2])
loss = layers.ReduceMean(in_layers=smce)
model.set_loss(loss)

output = layers.SoftMax(in_layers=dense2)
model.add_output(output)

model.fit(train_dataset, nb_epoch=10)
```

- model\_dir option
  - a directory where the model's parameters should be saved
- feature : 784(=28\*28) pixel 의 feature를 가짐
- label
  - 10 classes
  - vector is one-hot encoded
- None is used as an input dimension.
  - encodes the ability for a given layer to accept inputs that have any size in that dimension.
- Reshape
  - to convert our flat feature vectors into matrices of shape (28, 28)
- Conv2D
- Flatten
- ReduceMean
  - To average over all samples to obtain the final loss.
- Softmax
  - To transform the output with a SoftMax layer to obtain per-class output probabilities.
  - add this output to model with model.add\_output()

# SoftMax and SoftMaxCrossEntropy

- Softmax
  - You often want a model to output a probability distribution.
  - Every output must be positive, and they must sum to 1.
- SoftMaxCrossEntropy
  - first uses a softmax function to convert the outputs to probabilities,
  - then computes the cross entropy of those probabilities with the labels.
  - Remember that the labels are one-hot encoded: 1 for the correct class, 0 for all others.

```
>>> metric = dc.metrics.Metric(dc.metrics.accuracy_score)

>>> train_scores = model.evaluate(train_dataset, [metric])
>>> test_scores = model.evaluate(test_dataset, [metric])
```

# END