[Deep Learning for the Life Science - [O'Reilly](https://www.aladin.co.kr/shop/wproduct.aspx?ItemId=248232394&start=slayer)]

# Molecule data에 ML을 적용하는 방법

1. Molecular featurization (Molecule -> Vector) : Chemical Fingerprint, Chemical Descriptor
2. Graph Convolution Algorithms

# Featuring Molecular Data

1. Chemical Fingerprint

* 분자의 특성 유무를 bin(0/1)로 나타낸 vector 기반의 descriptor
* pros : bin -> 계산속도 빠름. 2개 분자의 유사도 측정 가능
* cons : 일부 정보 손실 문제 : 구조가 달라도 같은 fingerprint 발생하는 경우 있음

1. Chemical Descriptor

* 고전 물리학, 화학에 도움되는 통계량을 제공한다.
* pros : 상대적으로 분자의 일반적(고전적)인 특성에 의존하는 것을 예측할 때 유용함.
* cons : 원자의 상세한 배열에 의존하는 feature 계산에는 효과가 좋지 않다.

# GCN : Graph Convolutional Network

* 앞에 소개한 방법들은 사람이 설계한 방법이므로 ‘**ML model이 스스로 답을 찾아내는**’ 것이 불가능하다. 특정 통계량을 구하기 위한 방법들에 불과함.
* CNN : user가 어떤 pattern을 찾아야 하는지 알려주지 않고, model이 training을 통해서 스스로 pattern을 찾아낸다.
* Graph CNN : Element, charge(전하), 혼성화 등의 화학적 성질을 포함한다.
* Basic models : GraphConvModel, WeaveModel, **MPNNModel**, DTNModel
* cons : Graph 만으로 계산을 수행해 구조에 대한 정보 소실 -> 거대 분자에 비적합

[Kaggle solution - [MPNN](https://www.kaggle.com/code/widhiwinata/mpnn-a-type-of-graph-neural-network-gnn) : a type of GNN]

# Goal of Competition : predict BBB(Blood-Brain Barrier)

* BBB(0/1) : Membrane(얇은 막) separating the blood from the brain extracellular fluid
* Because of this, BBBP has been important to study for the development of new drugs that aim to the central nervous system

# Define features

1. AtomFeaturizer

* Generate a feature vector at each atom(node).
* These features can include the **chemical properties of atoms**.

1. BondFeaturizer

* Generate a feature vector for each **combination** including the type of bond, orientation of the bond, ring system, etc.

# Generate graphs

* Before generating complete graphs from SMILES, following functions are implemented.
* molecule\_from\_smiles(by RDKit) : SMILES(input) -> Molecule object(output).
* graph\_from\_molecule : Molecule object(input) -> three-tuple(output)
* ps. Splitting method : Scaffold splitting, Simple random splittings

# Create tf.data.Dataset

* In this tutorial, MPNN input will be take a single graph -> batch(merging) is needed
* **Global graph** : Merged & disconnected graph. : Each single subgraph is separated

# MPNN Model Baseline

* MPNN tutorial consists of three stage : message passing, readout and classification
* **Key** : MPNN **gradually update** the embedding of nodes by **exchanging** info between each node(atom) and edge(coupled) in the graph
* Process : Message Passing -> Readout Phase -> Fit to the model

1. Message passing

* Each node receives info from **neighboring** nodes and **updates** its status.
* The more repetitions, the more info it contains.

1. Readout Phase

* When the message passing procedure ends, the k(repetition)-step-aggregated node states are to be partitioned into subgraphs.
* Aggregate all nodes after message passing to generate graph-level embeddings.