DS-PBL I

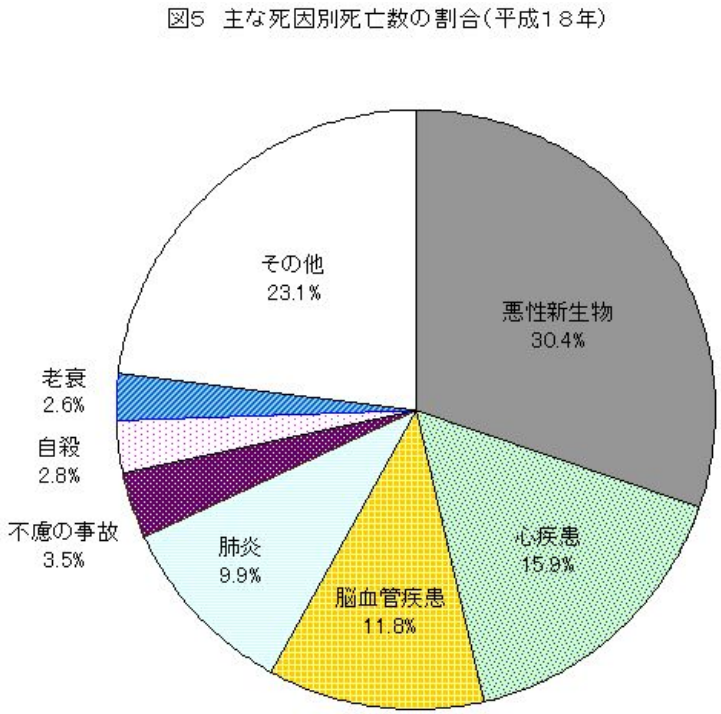
Prediction of active values for various compounds against a specific cancer cell by using machine learning
&
Search for effective molecule structures

Group5

Akihiro Kusuda, Kenta Izumi, Shunsuke Tanahashi
Takenari Nishihara, Yamato Mizushima

Backgrond

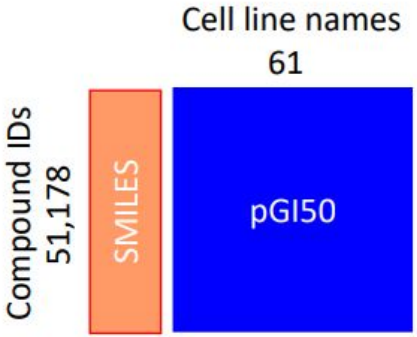
- According to statistics, cancer accounted for 30% of deaths in Japan as of 2006
- There are several ways to treat cancer, and chemical treatment is a typical method
- Predicting whether a compound is effective against cancer, or knowing what structures are effective against cancer, would be a great help in this regard



https://www.mhlw.go.jp/toukei/saikin/hw/jinkou/geppo/nengai06/kekka3.html

About the Data

- Cancer Cell Line Dataset
  - Compound structure and growth inhibition rate pair to cell lines



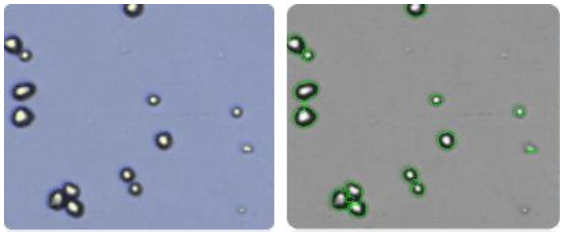
SMILES:
CN=C=O
[Cu+2].[O-]S(=O)(=O)[O-]
O=Cc1ccc(O)c(OC)c1
…

http://www-dsc.naist.jp/dsc_naist/wp-content/uploads/2022/07/2022_DS_PBL1.pdf

- pGI50 = negative logarithm of growth inhibition of 50%
  - The higher the value, the greater the effect.

Problem Setting

- Search for substances that inhibit the growth of cell lines
  - Regression to predict PGI50 with SMILES as explanatory variable
  - Binary classification to predict whether a compound is active or inactive based on its structure
- In this study, we specifically analyzed the activity of UACC-257



UACC-257

https://www.nexcelom.com/applications/cellometer/bright-field/nci-60-cancer-cell-lines/

Organizing Data

- Convert SMILES to Morgan fingerprints
  - o=C(O)CCCC(O)=O ⇒ $\mathbb{R}^{2048}$

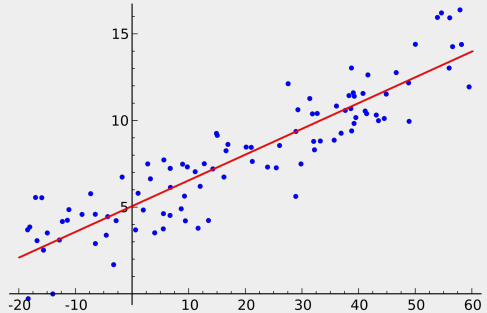
- Delete rows containing missing values
  - Delete the objective variable (activity value of UACC-257)

Method

Regression of pGI50

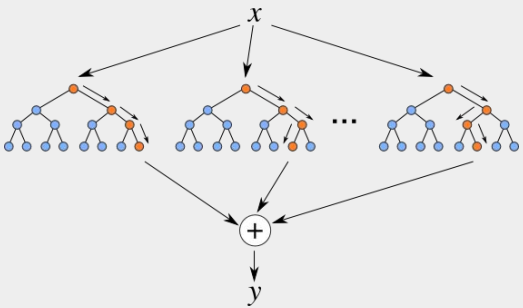
pGI50 >= 6 or not classified

linear regression



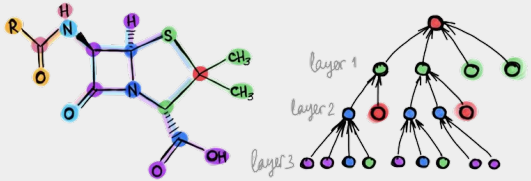
https://en.wikipedia.org/wiki/Linear_regression

random forest



https://levelup.gitconnected.com/random-forest-regression-209c0f354c84

graph convolutional NN

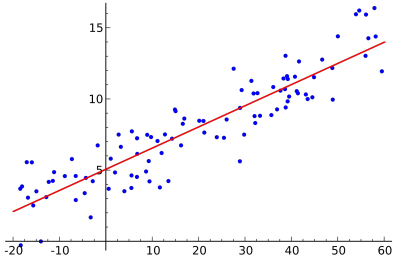


https://towardsdatascience.com/do-we-need-deep-graph-neural-networks-be62d3ec5c59

Method 1-1: linear regression·random forest

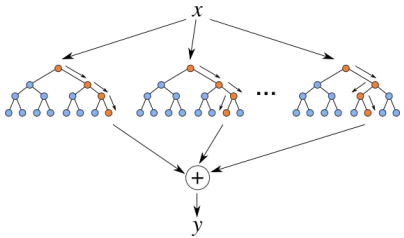
- linear regression:

  Y = w0 + w1*x1 + w2*x2 + … wn*xn



- random forest:

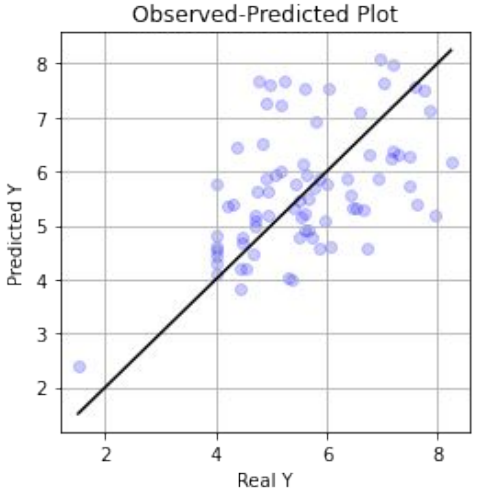
  Ensemble multiple decision trees



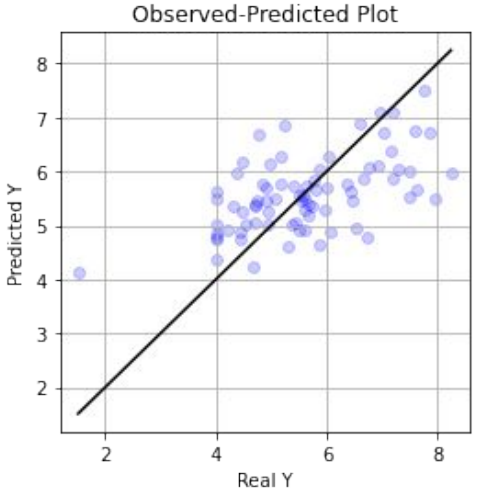
- Fingerprints to regression
  - Split data into train:test = 7:3 and evaluate

Result 1-1: linear regression･random forest

	R2	RMSE	MAE
linear regression	0.0725	1.18	0.939
random forest	0.323	1.01	0.801



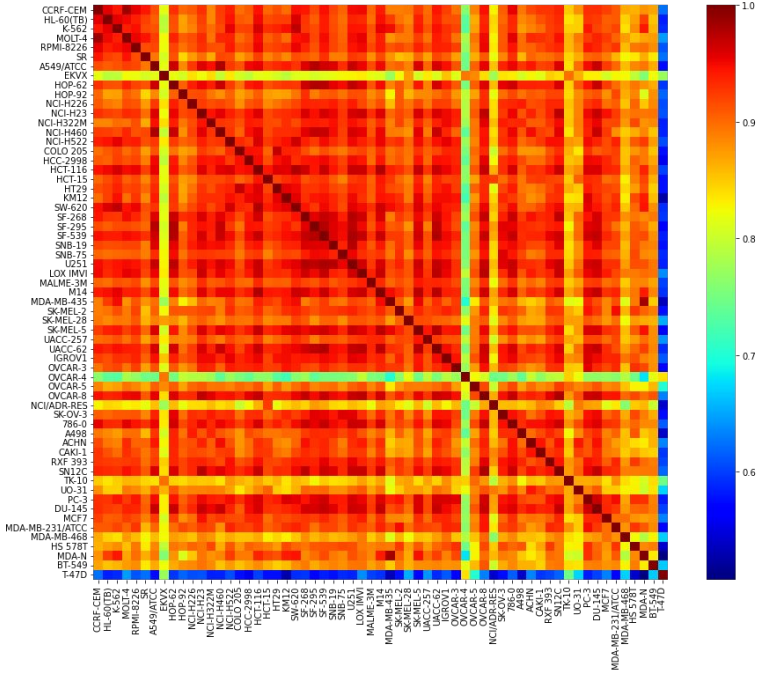
linear regression



random forest

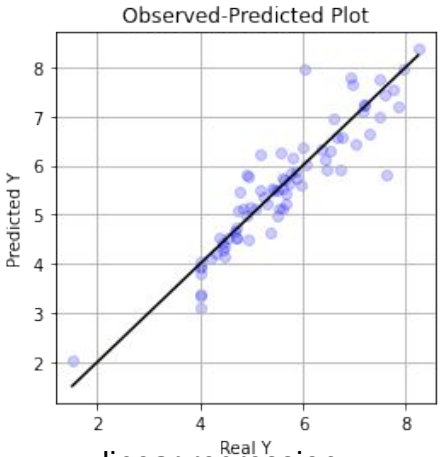
Method 1-2: linear regression·random forest

- Activity for most cell lines seems to correlate with otherwise activation sites
  - Fingerprints + other activation to regression

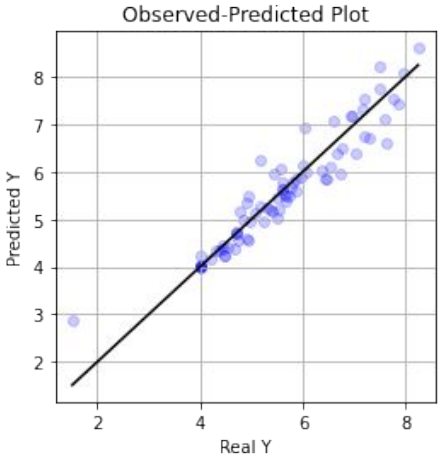


Result 1-2: linear regression･random forest

	R2	RMSE	MAE
linear regression	0.833	0.499	0.340
random forest	0.898	0.391	0.279



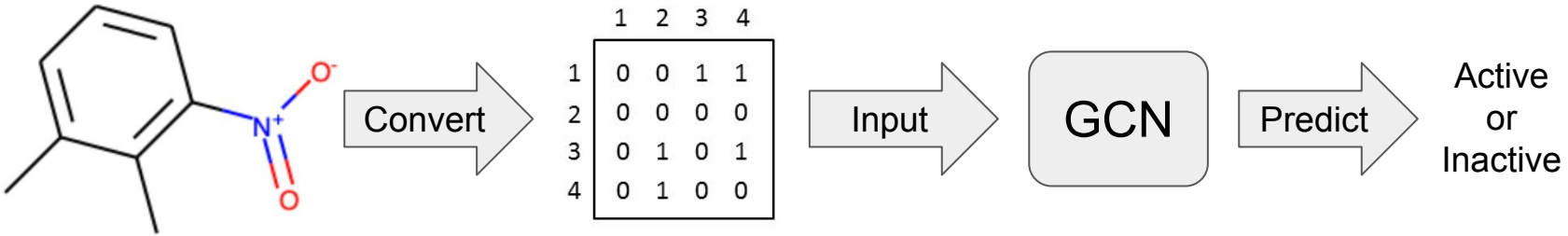
linear regression



random forest

Method 2: Graph Convolutional NN

- Binary classification of active or inactive with GCN

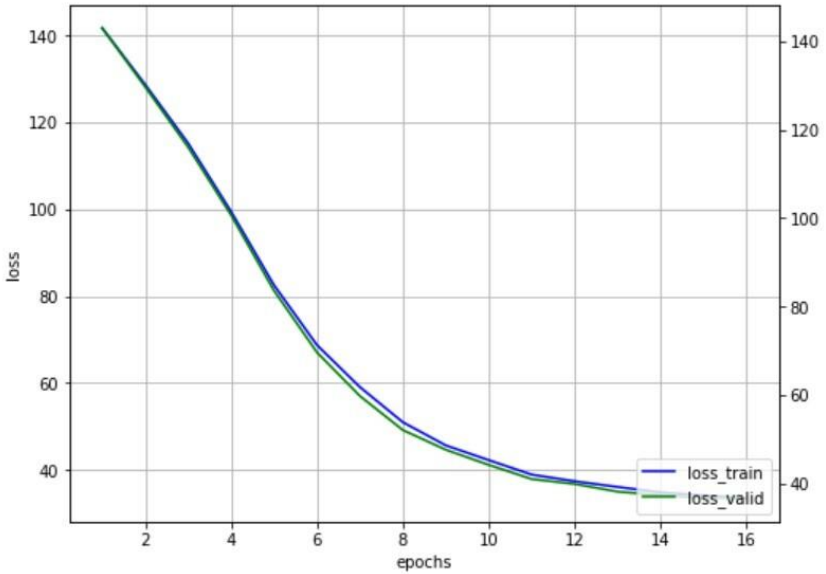


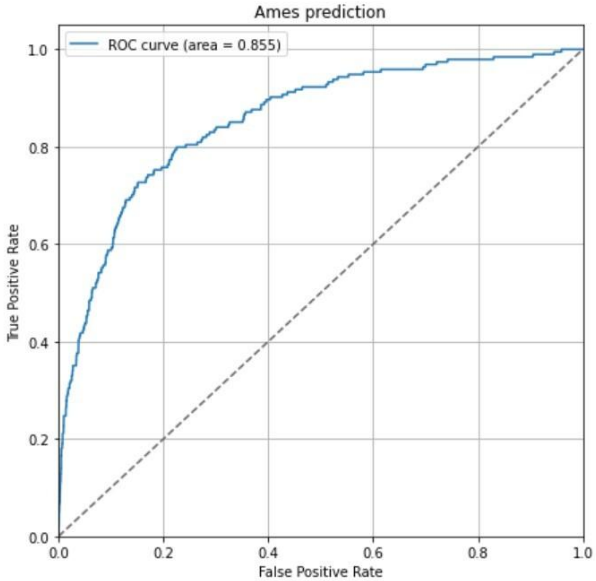
	1	2	3	4
1	0	0	1	1
2	0	0	0	0
3	0	1	0	1
4	0	1	0	0

- Model description:
  - GCNConv -> ReLU -> BatchNorm
  - (GCNConv -> BatchNorm) ×4
  - Global Add Pool
  - Linear -> ReLU -> BatchNorm -> DropOut
  - Log Softmax
- Settings:

  loss: cross entropy
  optimizer: Adam (lr=1e-4)
  split: stratified
  early stopping: patience=5

Result 2: Graph Convolutional NN

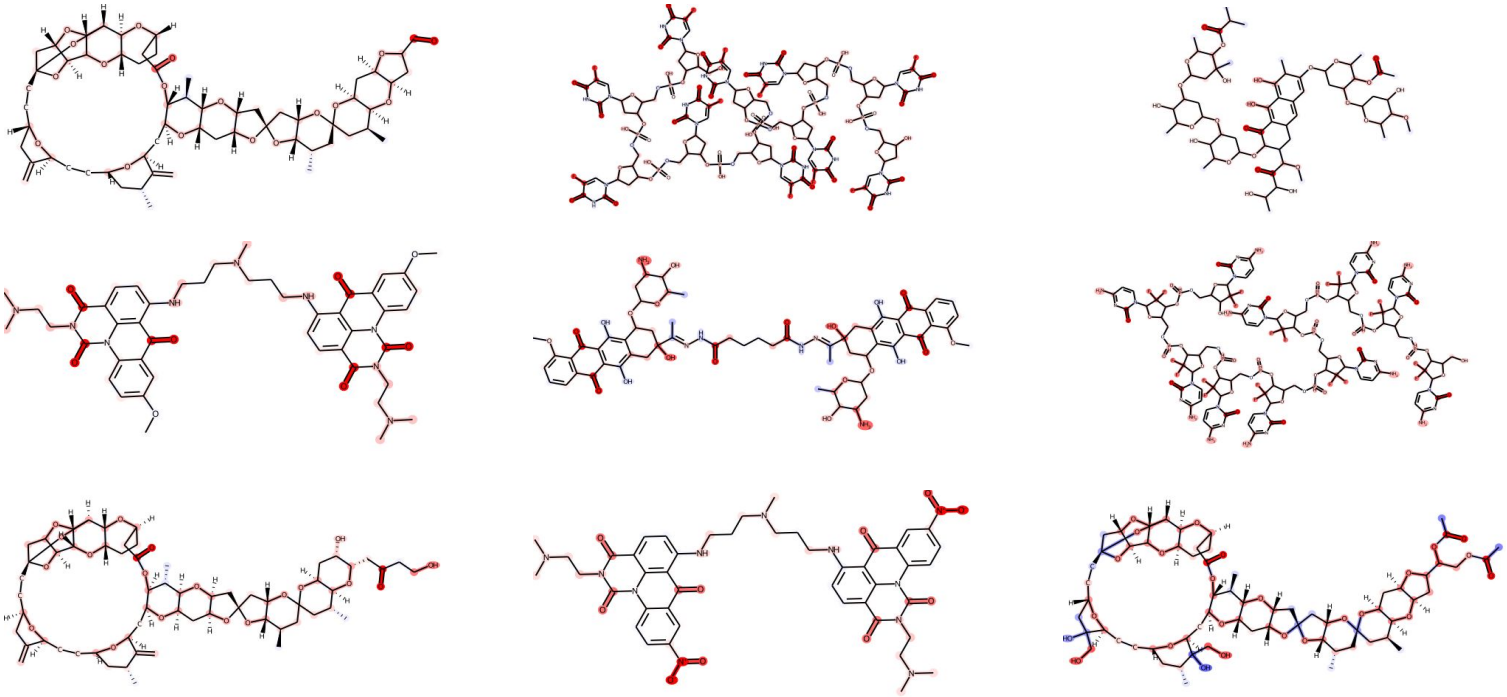




train auc: 0.875
test auc: 0.842

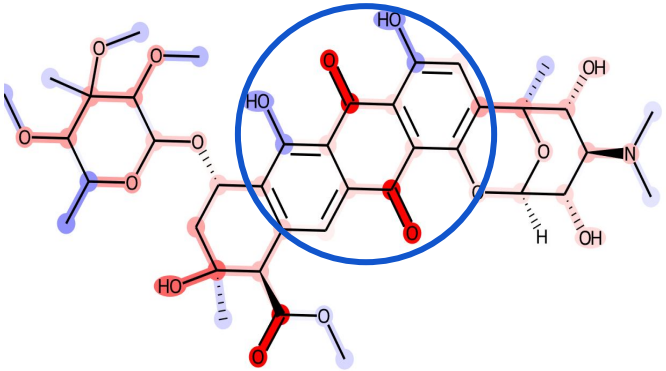
Result 2 Compounds predicted to be particularly active

- Visualize Integrated Gradient of edges https://arxiv.org/abs/1703.01365
- The red area contributes to the forecast.

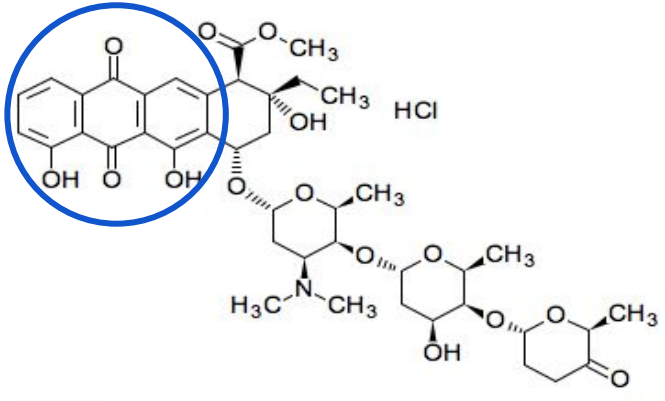


Discussion

Some of the compounds predicted in Result 2 showed structures similar to those of anticancer drugs.



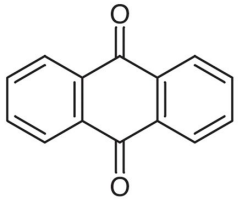
Compounds shown from result 2



D01911

Aclarubicin hydrochloride

https://www.genome.jp/dbget-bin/www_bget?dr_ja:D01911より改変

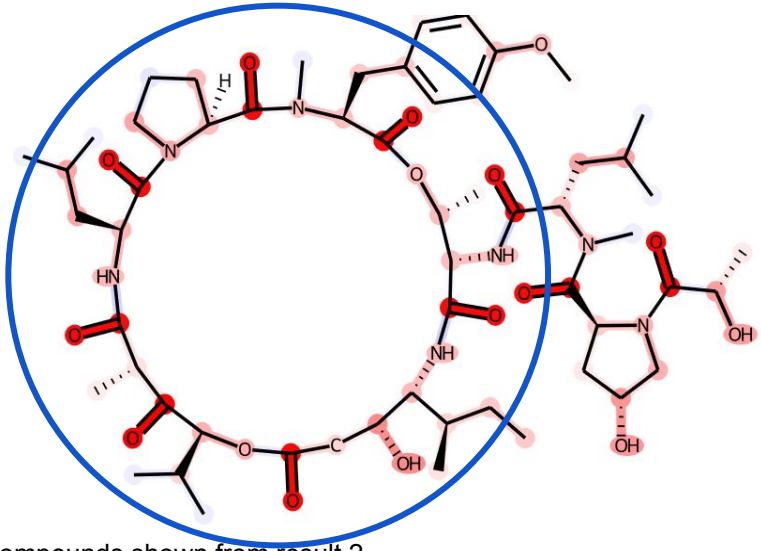


Antraquinone

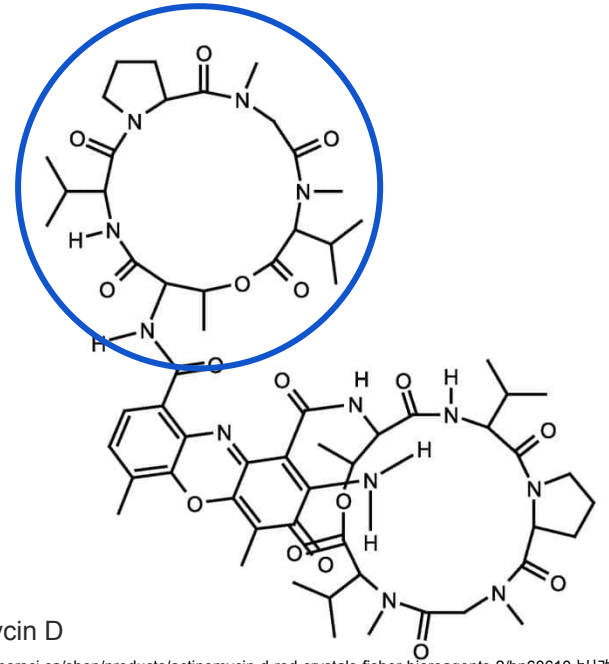
- 12 of the 30 predicted compounds had structures similar to Antraqunione
- Antraquinone inhibits DNA and RNA synthesis

https://gifu-pu.jpn.org/research/pcet/anthoraquinone

Discussion



Compounds shown from result 2



Actinomycin D

https://www.fishersci.ca/shop/products/actinomycin-d-red-crystals-fisher-bioreagents-2/bp60610より改変

- 15 of the 30 predicted results had macrocyclic structures.
- Macrocyclic compounds are abundant in nature as antibiotics.
  Some kind of interaction may have resulted in anticancer activity.

https://www.chem-station.com/blog/2015/04/discoveryofnewanticanceragents.html#:~:text=%E4%BE%8B%E3%81%88%E3%81%B0%E3%80%81%E5%A4%A7%E7%92%B0%E7%8A%B6%E3%83%A9,%E8%B3%AA%E3%81%AE%E4%BB%A3%E8%A1%A8%E6%A0%BC%E3%81%A7%E3%81%99%E3%80%82

Summary

- ➢ It is difficult to predict the activity of a substance from the Fingerprint alone
- ➢ The geometric structure of the compound was taken into account in the GCN to make predictions with good accuracy
- ➢ Anthraquinone and macrocyclic structures may be effective in suppressing cancer activity

Appendix

```
class MolecularGCN(torch.nn.Module):
    def __init__(self, dim, n_conv_hidden, n_mlp_hidden, dropout):
        super(MolecularGCN, self).__init__()
        self.n_features = 75 #  This is the mol2graph.py-specific value
        self.n_conv_hidden = n_conv_hidden
        self.n_mlp_hidden = n_mlp_hidden
        self.dim = dim
        self.dropout = dropout
        self.graphconv1 = GCNConv(self.n_features, self.dim, cached=False)
        self.bn1 = BatchNorm1d(self.dim)
        self.graphconv_hidden = ModuleList(
            [GCNConv(self.dim, self.dim, cached=False) for _ in range(self.n_conv_hidden)]
        )
        self.bn_conv = ModuleList(
            [BatchNorm1d(self.dim) for _ in range(self.n_conv_hidden)]
        )
        self.mlp_hidden =  ModuleList(
            [Linear(self.dim, self.dim) for _ in range(self.n_mlp_hidden)]
        )
        self.bn_mlp = ModuleList(
            [BatchNorm1d(self.dim) for _ in range(self.n_mlp_hidden)]
        )
        self.mlp_out = Linear(self.dim, 2)

    def forward(self, x, edge_index, batch, edge_weight=None):
        x = F.relu(self.graphconv1(x, edge_index, edge_weight))
        x = self.bn1(x)
        for graphconv, bn_conv in zip(self.graphconv_hidden, self.bn_conv):
            x = graphconv(x, edge_index, edge_weight)
            x = bn_conv(x)
        x = global_add_pool(x, batch)
        for fc_mlp, bn_mlp in zip(self.mlp_hidden, self.bn_mlp):
            x = F.relu(fc_mlp(x))
            x = bn_mlp(x)
            x = F.dropout(x, p=self.dropout, training=self.training)
        x = F.log_softmax(self.mlp_out(x), dim=-1)
        return x
```