How GCN “Sees” Chemical Toxicity

Chi Zhang
518021910395
Shanghai Jiao Tong University
iskyzh@sjtu.edu.cn

1 Introduction

People usually view deep nerual network as a blackbox. They just feed features into the DNN and it would automatically work. This is rediculous, especially for fields like medical. How could DNN tell for sure why a chemical is toxic? How much could we believe in DNN? In this report, we first analyzed feasibility of using DNN to classify chemicals, and then built and evaluated a model that classifies whether a chemical is toxic. We try to explain what DNN has “seen”, to convince ourselves this DNN really works.

2 Toxic or Not, This is a Question

When we are feeding something into the neural network, we must first ensure ourselves, is it reasonable to do this?

Luckily, telling if a chemical is toxic could be reasonable and explainable.

2.1 Chemical and SMILES

SMILES (Simplified molecular-input line-entry system) represents a chemical as a string. Given a SMILES, we could reconstruct the chemical structure. For example, given `OC1=C(Br)C=C(Br)C=C1C(=O)NC2=CC=C(Br)C=C2`, we could construct the chemical as seen in Figure 1.

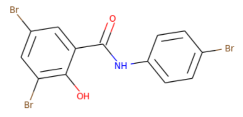


Figure 1: Chemical structure of `OC1=C(Br)C=C(Br)C=C1C(=O)NC2=CC=C(Br)C=C2`
This image is generated with http://hulab.rxnfinder.org/smi2img

2.2 How, Toxic?

Before we feed **SMILES** into the neural networks, we must ask ourselves, will that really work?

First of all, we should define toxic. According to MoleculeNet: A Benchmark for Molecular Machine Learning, we have a lot of datasets. We classify chemicals by their toxicity to different targets of human beings. For example, in Tox 21, we have `NR-ER,NR-ER-LBD`, etc., which are different human biological targets.

Are there any properties that can be learned by human beings, or neural networks? Luckily, people find ways to recognize whether a chemical is toxic or not. Let's take a look at Figure 2. `CC=CCC#N` is not toxic, and `NCGC00163509-01` is toxic. How a chemical affects human is closely related to its functional group. `CC=CCC#N` only contains simple function groups involving `C` and `N`, so it has a higher probability of not being toxic. On the contrast, `NCGC00163509-01` contains `Cl` on Benzene ring. Therefore, it is highly possible that it is toxic.

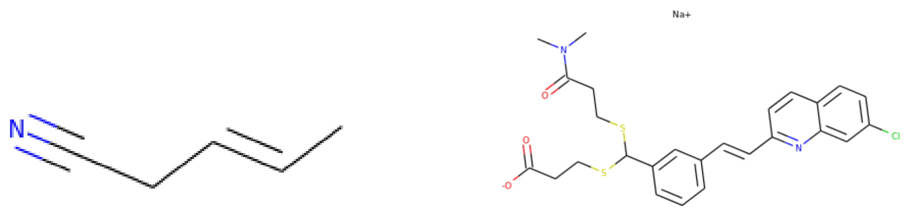


Figure 2: Chemical structure of `CC=CCC#N` (left) and `NCGC00163509-01` (right)

2.3 Represent Features in Neural Network

Properties of a chemical is closely related to its functional group. How to represent such "function group" in neural network?

A function group can be seen as a graph. A vertex represents an atom, and an edge in graph indicates that there is chemical bond between two atoms. By representing such group as a graph structures, neural networks could easily learn properties of a chemical.

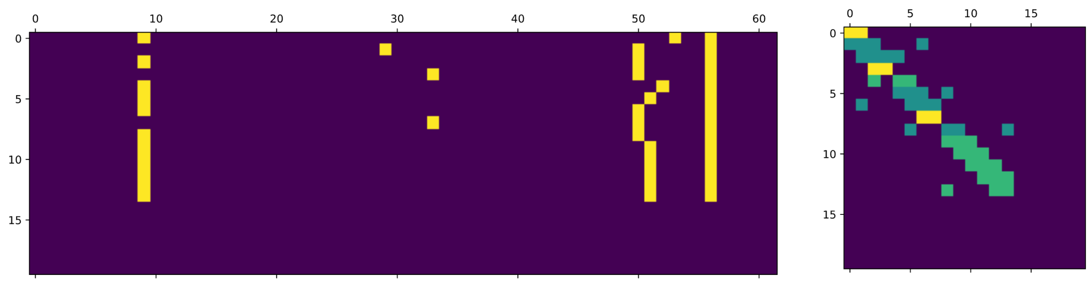
To encode an "atom" in a vertex, we could use one-hot encoding scheme. We use a vector to represent one vector. Each element of a vector is either 0 or 1. These features include:

Element of an atom. For example, if the atom is `C`, then the corresponding element of vector is 1.

Number of `H` atoms around this atom.

Charge of atom. For example, -6, $+4$. Different charge occupies one place in vector.

By combining features of all vertices, we could obtain a feature matrix and an adjacent matrix of a chemical.



Feature Matrix

Adjacent Matrix

Figure 3: Feature Matrix of Chemical

Layer	Parameter	Activation
Input	62 features with 132 vertices	
Graph Convolution 1	128 features	ReLU
Graph Convolution 2	256 features	ReLU
Graph Convolution 3	512 features	ReLU
Global Mean Pool		
Dense	512	Sigmoid
Dense	128	Sigmoid
Dropout	0.25	
Dense	2	Softmax

Table 1: Model Parameters

Note that before feeding adjacent matrix into the neural network, we need to do some normalization. Refer to source code for more information.

3 Simple Model could be Powerful

3.1 GCN Implementation in Tensorflow

Tensorflow could do complex graph optimization, and thus boost train efficiency. By leveraging Tensorflow Keras API, we could build and train a neural network with ease.

A GCN layer can be formulated as Equation 1.

$$H^{(l+1)} = \sigma(\tilde{A}H^{(l)}W_1^{(l)} + W_0^{(l)}) \tag{1}$$

Where $\tilde{A}$ is the adjacent matrix of a graph, $H^{(l)}$ being the input, $H^{(l+1)}$ being the output, $W_1^{(l)}$ being the weight, and $W_0^{(l)}$ being the bias. Note that $H^{(0)}$ is the feature matrix we mentioned before. In this case, it should be 132×62. (We have 62 one-hot-encoded features, and pad every chemical to have the same vertex number 132.)

In case of Tensorflow Keras, the GCN layer should be able to compute in batch. Therefore, the input to GCN layer is two matrices, with size of $\mathrm{batch} \times 132 \times 62$ and $\mathrm{batch} \times 132 \times 132$ respectively. And we manipulate them as Equation 1, to get the output.

3.2 Full GCN Model

As seen in Figure 4 and Table 1, the final model is composed of several GCN layers, followed by dense layer.

GCN layers help recognize graph features in chemical. As there are a lot of carbon rings in chemicals, and generally, 6 carbons form a ring, GCN depth of 3 can best recognize this structure.

Features should have no relation with which vertex it lies on. For example, `CO` should have the same feature of `OC`. Therefore, after GCN layers, we use a global mean pool to summarize one feature.

And then, we use dense layer to further learn relationships between features. Finally, they are feed into a softmax layer, with each neuron representing how toxic (not toxic) a chemical is.

3.3 What does DNN see

Knowing what DNN has learned is important in medical field. In this section, we try to explain what DNN has seen, and prove effectiveness of our network structure.

As seen in Figure 5, we can observe the following properties:

Weights in one row has nearly the same value. This means that vertex features have no relationship with their position, which is as expected.

Only a few “composed” features have effects on final output. Deeper layers have fewer useful features.

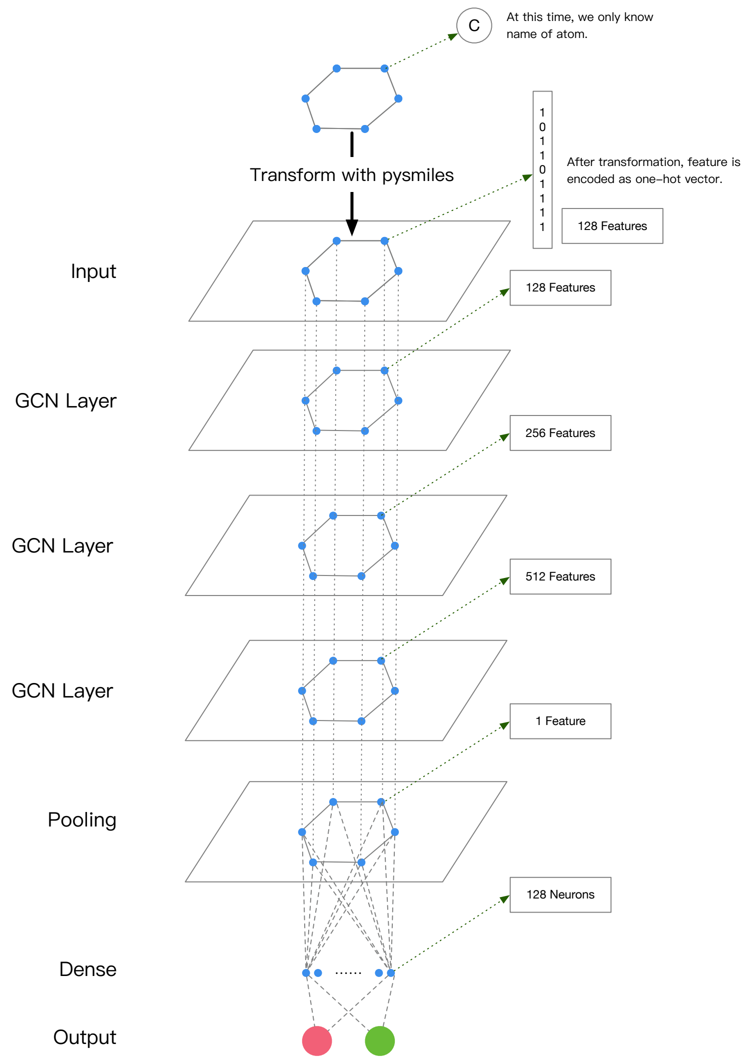


Figure 4: Full GCN Model

This makes us believe that our DNN learns some patterns from chemicals.

3.4 Tricks

We resampled both the train set and validation set, to make positive and negative samples are of the same number. One train epoch will visit all negative data, but not all positive data.

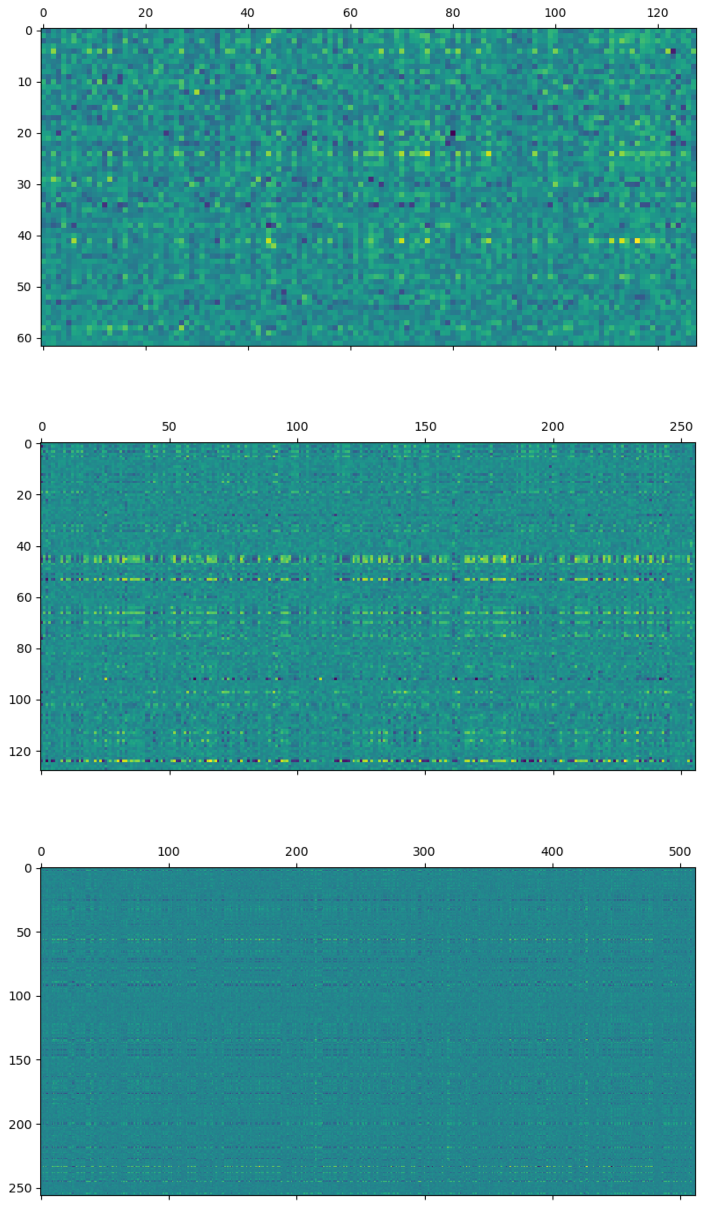


Figure 5: Weights of GCN layers (Top to Bottom)

3.5 Evaluation Result

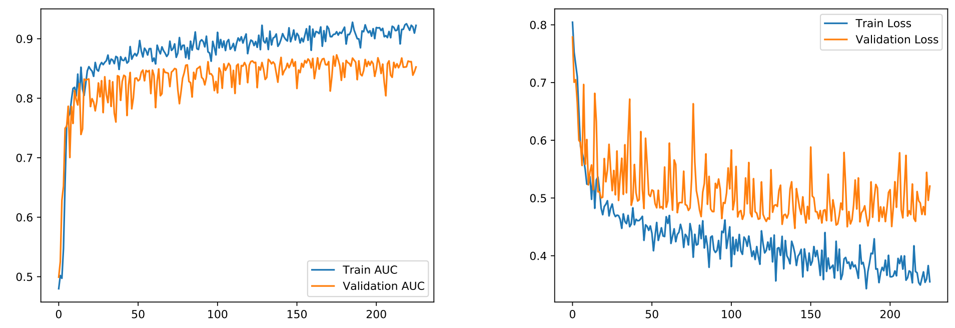


Figure 6: Evaluation of GCN Model

This model gets an AUC of 0.90115 on Kaggle.

4 Adding Complexity makes No More Power

4.1 Adding More Layers

Using more layers and neurons doesn't necessarily yield better result. In this experiment, I added a fourth GCN layer. This makes no sense, probably because 3 layers is powerful enough to extract necessary features for carbon-based chemicals.

4.2 Adding More Features

Adding more features doesn't contribute to better AUC either. It seems that we have extract enough information from chemical atoms. To get a better accuracy, I suppose we could:

Extract some "union" features and higher-order. For example, whether a `C` atom is the neighbour of `S`. We can construct such composed features. Embedding such feature group into one feature may make the DNN learn better.

Use more data. It seems that most chemical-related datasets have multiple learning targets. By leveraging multi-task learning, we may get a better result.

4.3 Evaluation Result

I've played with dozens of different network structures, and here we choose one of the most complex model among them.

First of all, we simply add the fourth GCN layer.

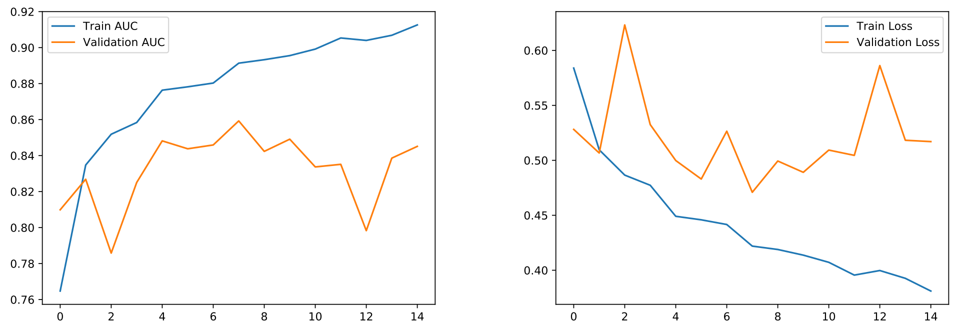


Figure 7: Model with 4 GCN Layers

It turns out that this model would easily overfit on train set.

Then, we extract more fine-grained features from chemicals. We view electron `+1` and `-1` as different features, and expand features related to `H` atom.

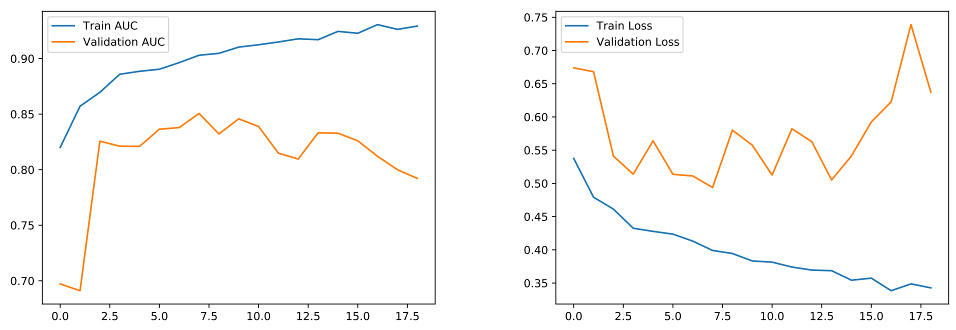


Figure 8: Model with More Features

This model also suffers from overfit.

Therefore, more layers and more features don't necessarily lead to a better model. We should always try to "explain" our features and models, to get a best-fit model for a specific task.

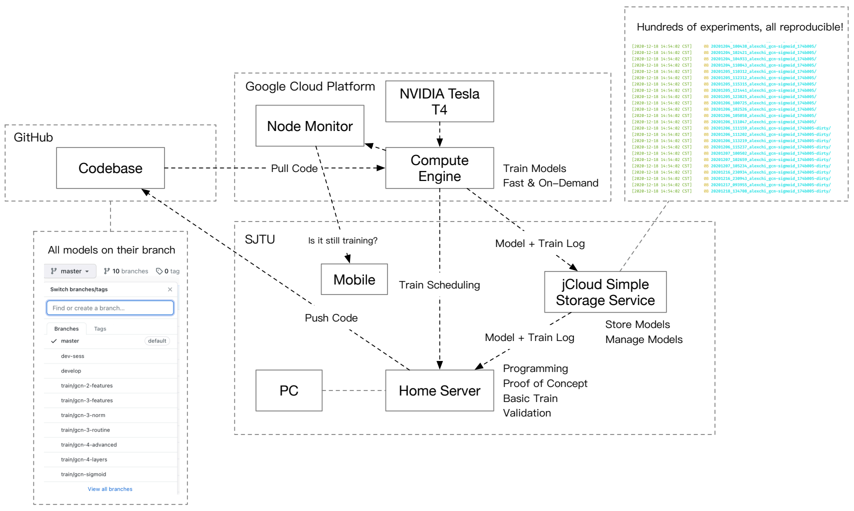
5 Machine Learning Infrastructure

5.1 Speed up Data Pipeline

By using Tensorflow Dataset API, we reduce time to first epoch (TTFE) to only 3 seconds. We do not need to wait all chemicals to be loaded and processed by pysmiles before training. They are automatically pre-fetched and cached by Tensorflow.

This change significantly improved model training and evaluation workflow, as now we could know how a model performs sooner.

5.2 Cloud-based ML Workflow



```
[2020-12-18 14:54:02 CST]     0B 20201204_100438_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201204_102421_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201204_104933_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201204_110843_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201205_110312_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201205_112312_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201205_115315_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201205_121441_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201205_123825_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201206_100725_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201206_102526_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201206_105058_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201206_111047_alexchi_gcn-sigmoid_174b005/
[2020-12-18 14:54:02 CST]     0B 20201206_111159_alexchi_gcn-sigmoid_174b005-dirty/
[2020-12-18 14:54:02 CST]     0B 20201206_111202_alexchi_gcn-sigmoid_174b005-dirty/
[2020-12-18 14:54:02 CST]     0B 20201206_113219_alexchi_gcn-sigmoid_174b005-dirty/
[2020-12-18 14:54:02 CST]     0B 20201206_115237_alexchi_gcn-sigmoid_174b005-dirty/
[2020-12-18 14:54:02 CST]     0B 20201207_100502_alexchi_gcn-sigmoid_174b005-dirty/
[2020-12-18 14:54:02 CST]     0B 20201207_102659_alexchi_gcn-sigmoid_174b005-dirty/
[2020-12-18 14:54:02 CST]     0B 20201207_105234_alexchi_gcn-sigmoid_174b005-dirty/
[2020-12-18 14:54:02 CST]     0B 20201216_230934_alexchi_gcn-sigmoid_174b005-dirty/
[2020-12-18 14:54:02 CST]     0B 20201216_230943_alexchi_gcn-sigmoid_174b005-dirty/
[2020-12-18 14:54:02 CST]     0B 20201217_093955_alexchi_gcn-sigmoid_174b005-dirty/
[2020-12-18 14:54:02 CST]     0B 20201218_134708_alexchi_gcn-sigmoid_174b005-dirty/
```

```
✓ master    default
dev-sess
develop
train/gcn-2-features
train/gcn-3-features
train/gcn-3-norm
train/gcn-3-routine
train/gcn-4-advanced
train/gcn-4-layers
train/gcn-sigmoid
```

Figure 9: Machine Learning Workflow

By leveraging several cloud services, we take track of all models and make all of them reproducible. Every model we have produced are stored, and can be evaluated. As seen in Figure 9, programming is mainly done in local server, and model is trained on Google Cloud Platform. After training, models are saved in SJTU S3 service, and transferred back to local server for evaluaion. Each model has a corresponding codebase, and they are stored on GitHub as branch. By making full use of such ML infrastructure, we managed to train and evaluate 200 models seamlessly.

6 Conclusion

In this project, we succeeded to build a GCN model to identify whether a chemical is toxic or not, with an AUC of 0.90115 on test dataset. We also explored other network structures, and found that more layers and more features doesn't necessarily contribute to a better model. By using a cloud-based machine learning infrastructure, we make every model reproducible.

I would like to thank TA Qizhi Li for his help. He spot several errors in my early version of model that makes the result very bad. After fixing these errors, the AUC rocketed onto 84%. After that, I proposed my own GCN model and got even better result.

A 200 Models