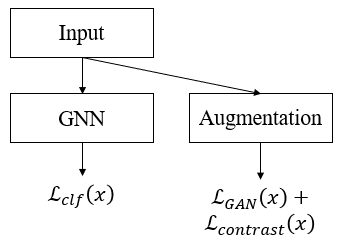
Current Objective

* Reduce Flood Risk with Soil Dataset
  + Use B’s information when prediction A (water flow)
    - Directed GNN
  + Few labels
    - Augmentation-based contrastive learning

Algorithm

1



1. Feature Processing

2

3

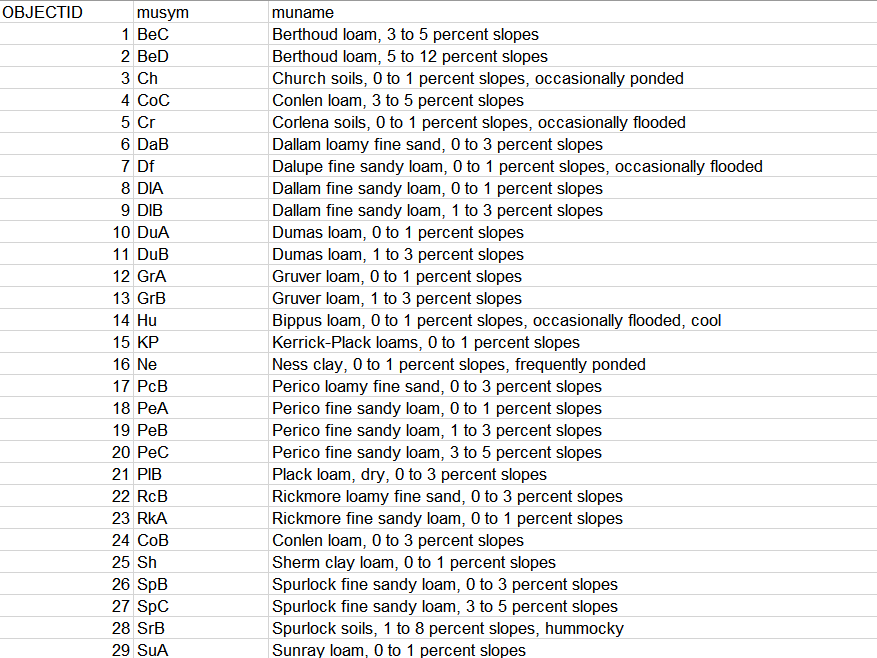
1. GNN (Neighbor Feature Aggregation)
2. Contrastive Learning with Aggregated Feature

Loss Function

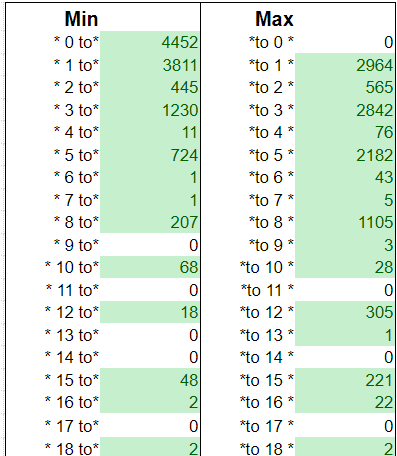
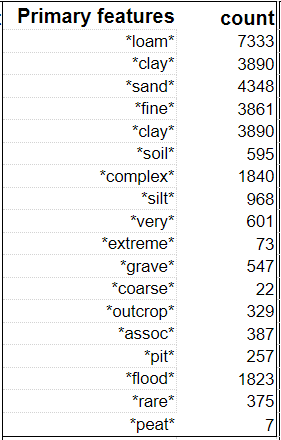
1. GNN:
2. GAN:
3. Contrast:

Overall loss

Problems for Feature Processing

* One of the features has high dimension.
* 

**3,306 types**



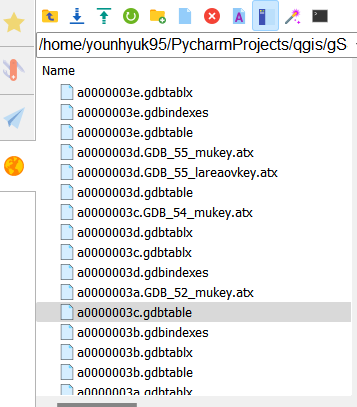
**weight**

**18 features**

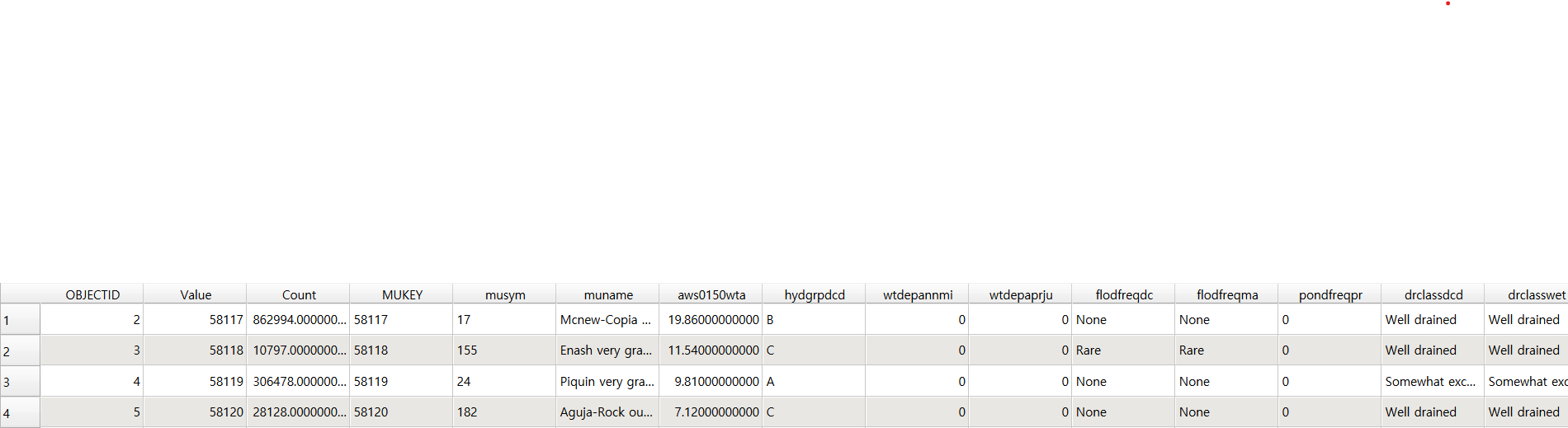
Normalize?

Embedding

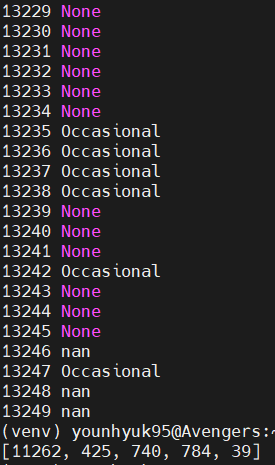
(Task 1) finding locations with low flood risk using soil dataset

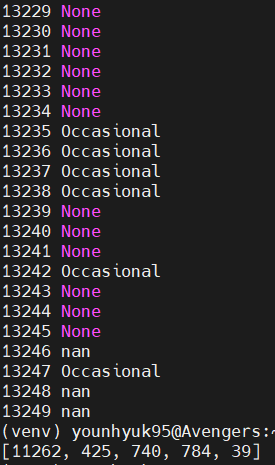


Dataset 🡪 a0000003c.gdbtable



13250 grids 🡪 Train (80%) / Valid (10%) / Test (10%) split 🡪 10,600 / 1,325 / 1,325.

Should split the dataset considering the flood frequency labels.

**[None, Rare, Occasional, Frequent, Very Frequent]**

After split (training sample)



Data preprocessing



Continuous

Discrete

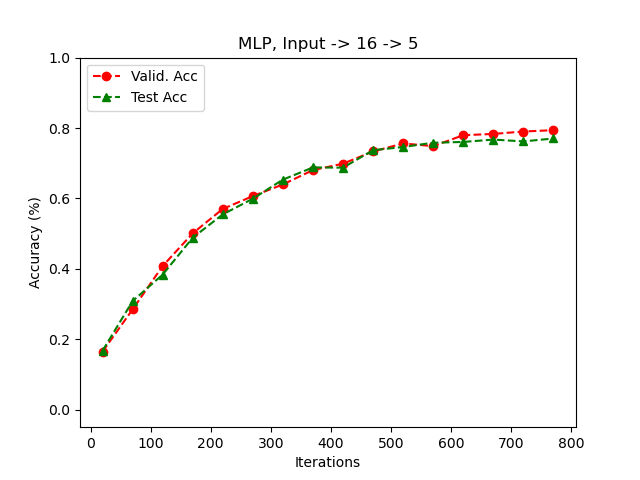
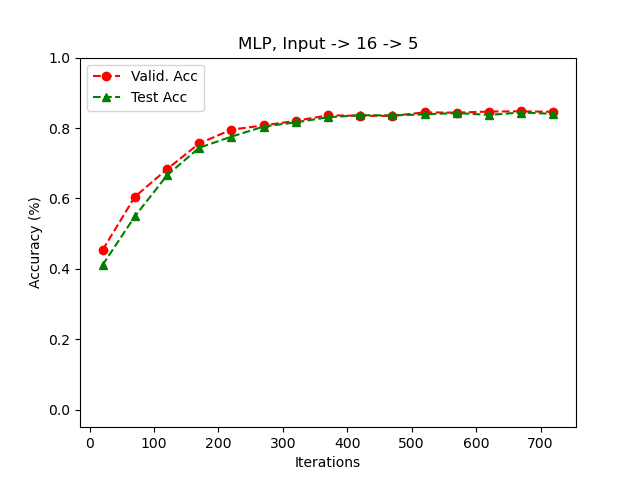
Implementing plane MLP

Train (80%) / Valid (20%) / Test (20%) split 🡪 10,600 / 1,325 / 1,325

Features: 32 D & Class: [None, Rare, Occasional, Frequent, Very Frequent]

Validation / Test Acc. (%) for every 50 epochs

* Use early stopping (if valid. Acc. doesn’t improve for 100 epochs)



Test Accuracy: 84.78 %, Layer: 32 🡪 32 🡪 5

Test Accuracy: 83.27 %, Layer: 32 🡪 16 🡪 5

2/5 (Mon) – **Class Imbalance**

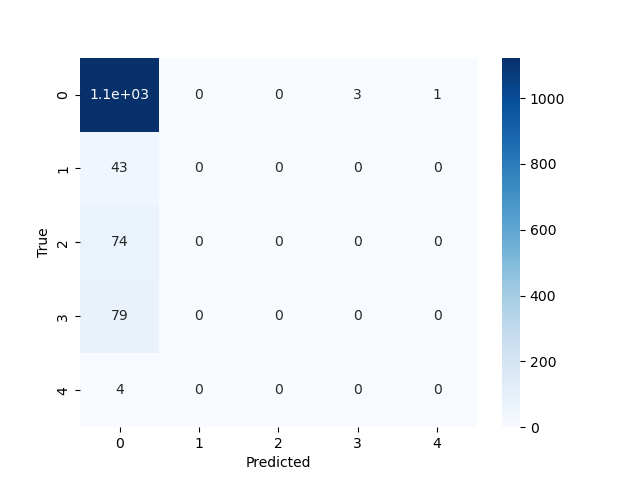
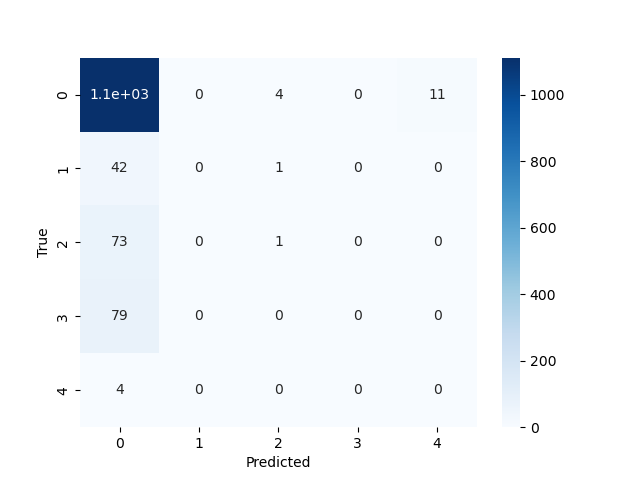
Confusion matrix (class prediction ratio)

* Dr. Candan, Paras

Finding a solution to handle the class imbalance problem

02/07 (Wed) – Confusion matrix

Test data: 1,325 samples (confusion matrix)

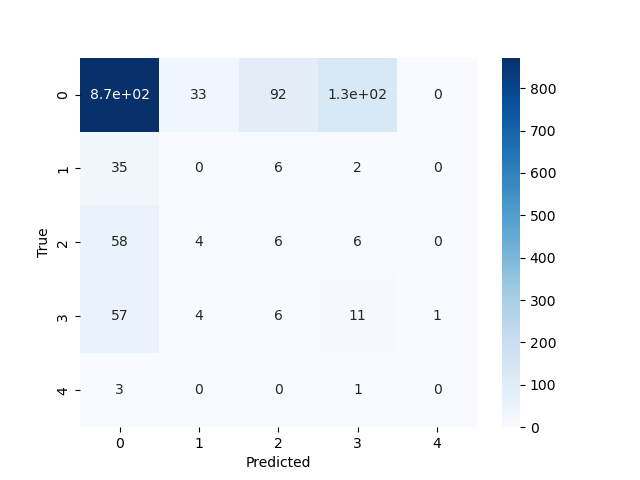


Test Accuracy: 84.5 %

Test Accuracy: 83.3 %

02/09 (Fri) – Loss weight adjustment

Example, training sample distribution



A = [85%, 3.2%, 5.6%, 5.9%, 0.3%]

Loss adjustment 🡪 1-A

Test Accuracy: 78.6 %

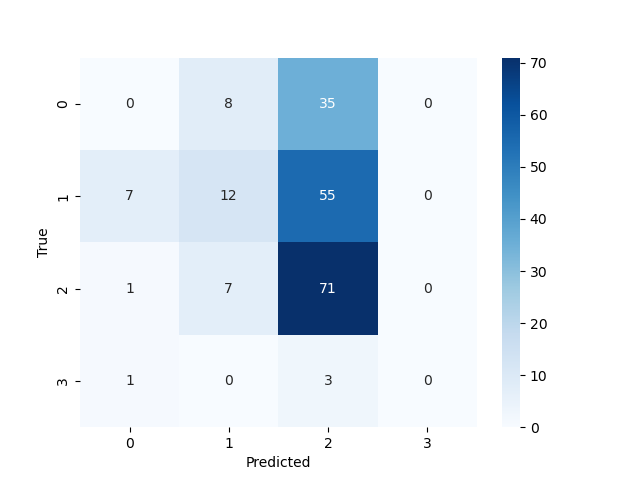
02/13 (Tue)

Exclude class ‘None’ from samples.

🡪 [Rare: 340, Occasional: 592, Frequent: 627, Very Frequent: 31]

**Prediction Acc**: **41.5 %**

[Rare: 9, Occasional: 27, **Frequent: 164,** Very Frequent: 0]

****

Very Frequent

Frequent

Rare

Occasional

True labels: [Rare: 43, Occasional: 75, **Frequent: 78**, Very Frequent: 4]

Solutions

1. Separate ‘None and Rare’ ‘Occasional, Frequent, Very Frequent’
   1. Finding the area with low flood risk (not optimal solution)
2. Balanced sampling



**600**

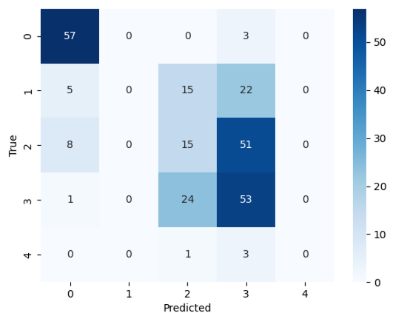
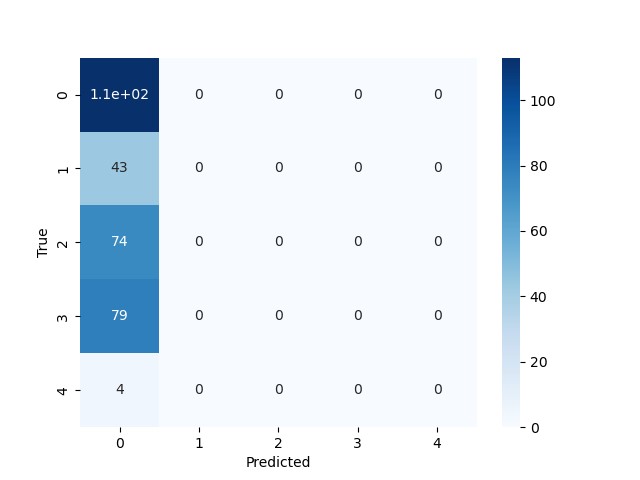
* 1. Reduce the number of ‘None’ class.

w/o class adjustment

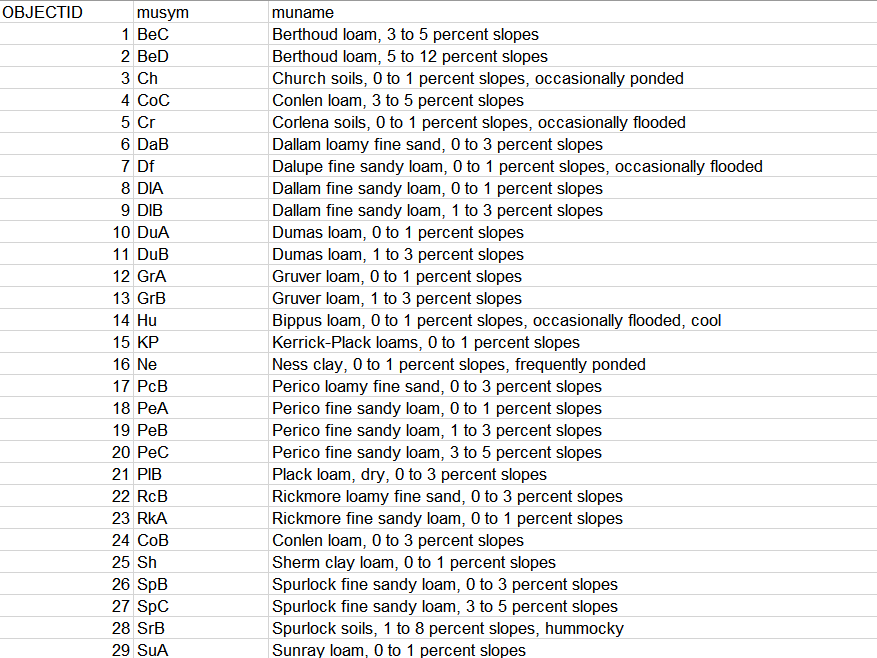
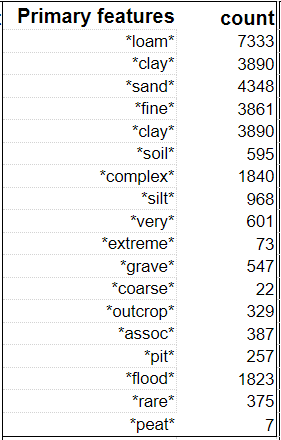
Accuracy: 36.1 %

w/ class adjustment

Accuracy: 46.5 %



2/16 (Fri) Use Soil element (vectorization).



🡪 [1, 0, 0, 0, …, 0, 0]

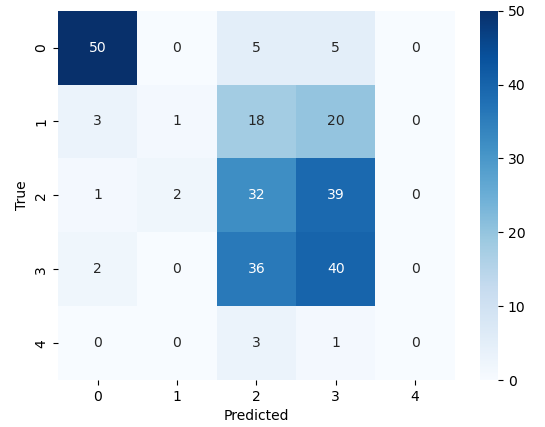
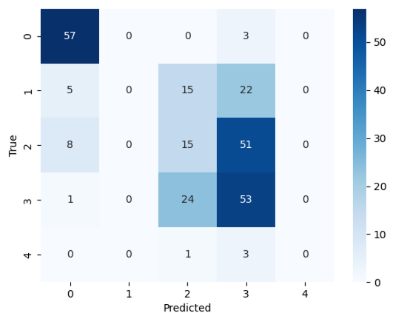
Vectorize



**600**

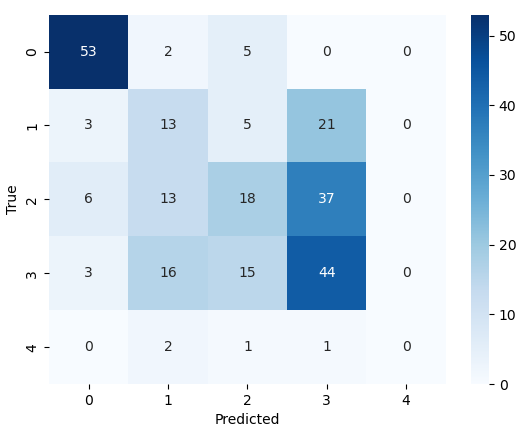
w/o soil feature

Accuracy: 46.5 %



w/ soil feature

Accuracy: 47.7 %

**w/ soil feature, adaptive sampling, learning ratio adjustment.**

Accuracy: 50.78 %

**2/23 (Fri)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **0** | **0** | **0** | **0** |
| **0** | **1** | **0** | **0** | **0** |
| **0** | **0** | **1** | **0** | **0** |
| **0** | **0** | **0** | **1** | **0** |
| **0** | **0** | **0** | **0** | **1** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **0.8** | **0.5** | **0.25** | **0** |
| **0.8** | **1** | **0.6** | **0.3** | **0** |
| **0.25** | **0.5** | **1** | **0.8** | **0.6** |
| **0** | **0.4** | **0.8** | **1** | **0.8** |
| **0** | **0.3** | **0.6** | **0.8** | **1** |

[None, Rare] [Occasional, Frequent, Very Frequent]

similar

similar

**New Accuracy Measure (79%)**

**Previous Accuracy Measure (50%)**

Prediction

Prediction

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **0.8** | **-0.1** | **-0.2** | **-0.5** |
| **0.8** | **1** | **-0.1** | **-0.2** | **-0.5** |
| **-0.5** | **-0.2** | **1** | **0.5** | **0.2** |
| **-0.6** | **-0.2** | **0.4** | **1** | **0.4** |
| **-0.5** | **-0.2** | **0.2** | **0.5** | **1** |

**Normalized Accuracy Measure (56%)**

True

True

Hierarchical loss

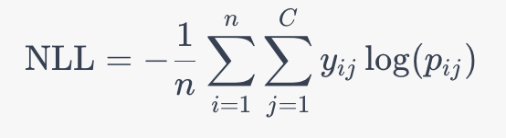
[None, Rare] [Occasional, Frequent, Very Frequent]

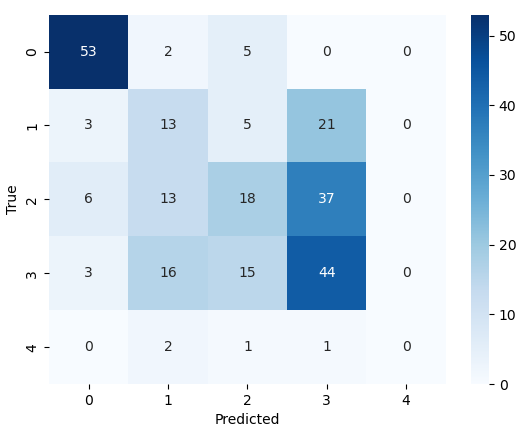
**small loss**

**small loss**

**large loss**

Accuracy: 50.78 %





and

and

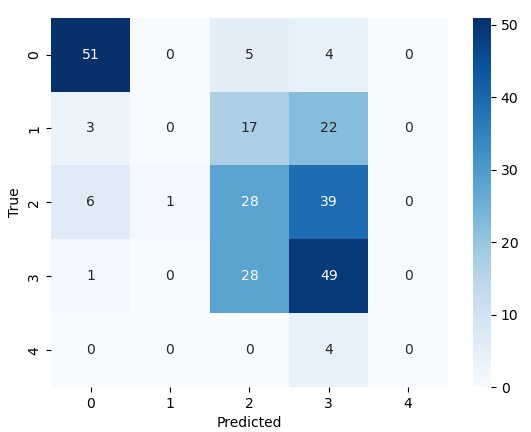
**In both cases, NLL loss has same value**

[None, Rare] [Occasional, Frequent, Very Frequent]

**Intra class**

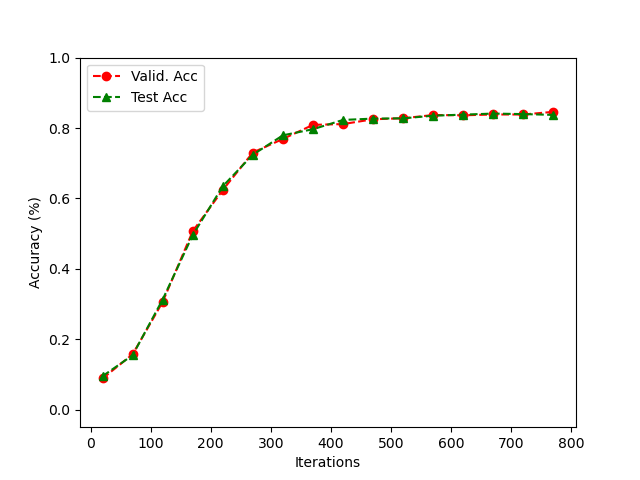
**Intra class**

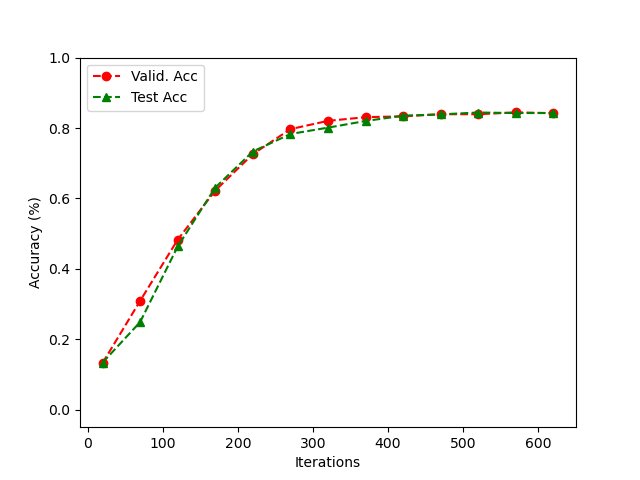
**Inter class**



Contrastive loss

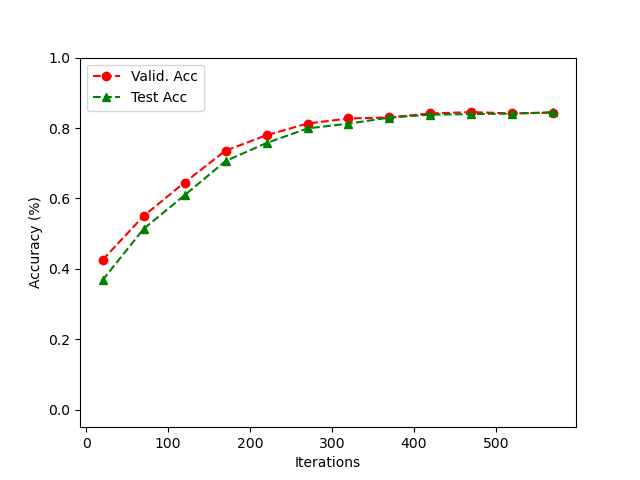
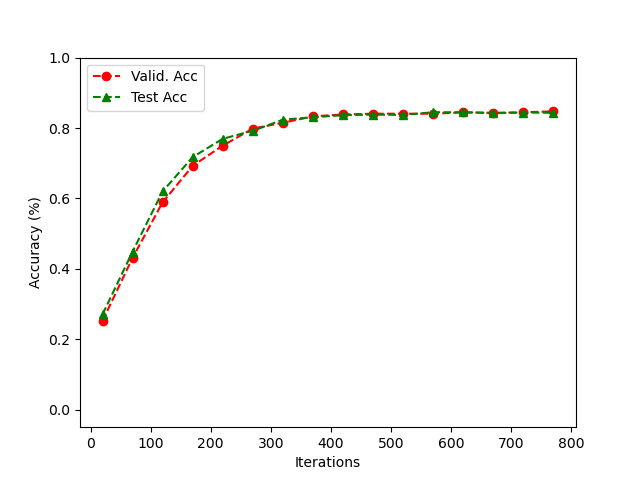
Accuracy: 50 ~ 54 %

**Adding gaussian noise**



Test Accuracy: 84.4 % (Noise: 0.01)

Test Accuracy: 83.57 % (Noise: 0.01)



Test Accuracy: 84.17 % (Noise: 0.05)

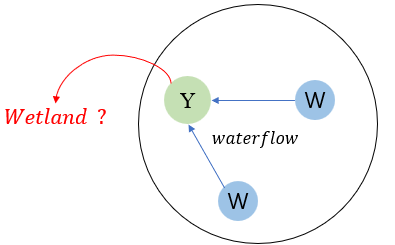
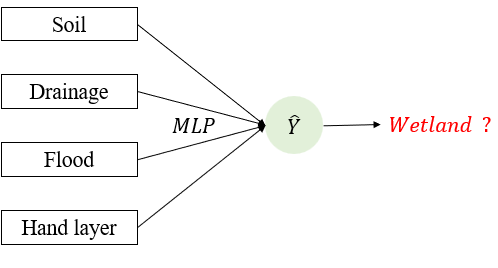
Test Accuracy: 84.85 % (Noise: 0.05)

Problems for Feature Processing

1. Discrete features 🡪 single value?

Ex) flood frequency 🡪 [None, Rare, Occasional, Frequent, Very frequent] 🡪 [0, 0.25, 0.5, 0.75, 1.0]

Problems for GNN (Information Propagation)

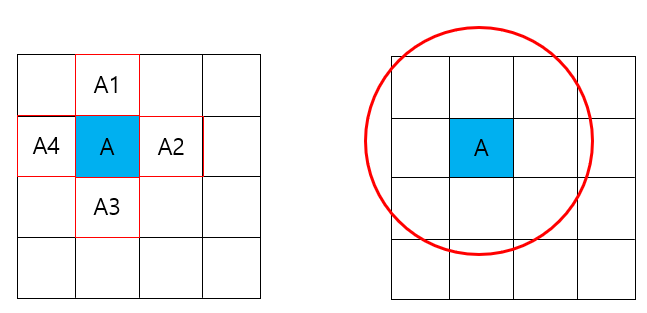


use neighbor

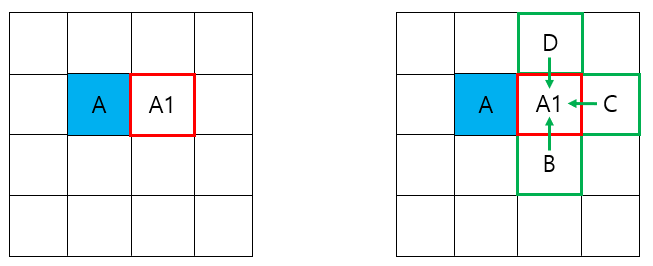
information

Directed GNN (direction of waterflow)

Plane MLP

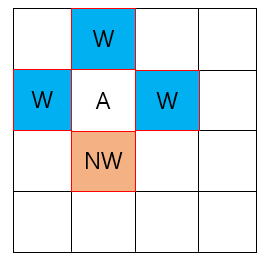
1. Range of neighbors

* Only the adjacent grids or set specific miles (e.g., within 10 miles)?



1. What if the neighbor’s feature doesn’t exist?

* Ignore them or apply data imputation?



1. Neighbors are not always homophilic (have different labels)

* How can we figure out this and propagate information?

Problems for Contrastive Learning

1. How to augment features?

* GAN vs Noise addition
  + 🡪 GAN
  + 🡪 Noise addition