kaggle

Kaggle Winner Presentation (5th)

NeurIPS 2024 - Predict New Medicines with BELKA

https://www.kaggle.com/competitions/leash-BELKA

HENG CHER KENG
https://www.kaggle.com/hengck23

Agenda

Agenda

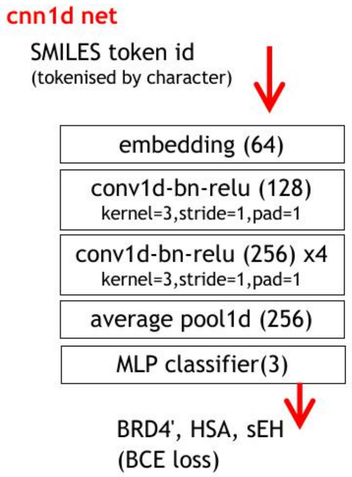
Background

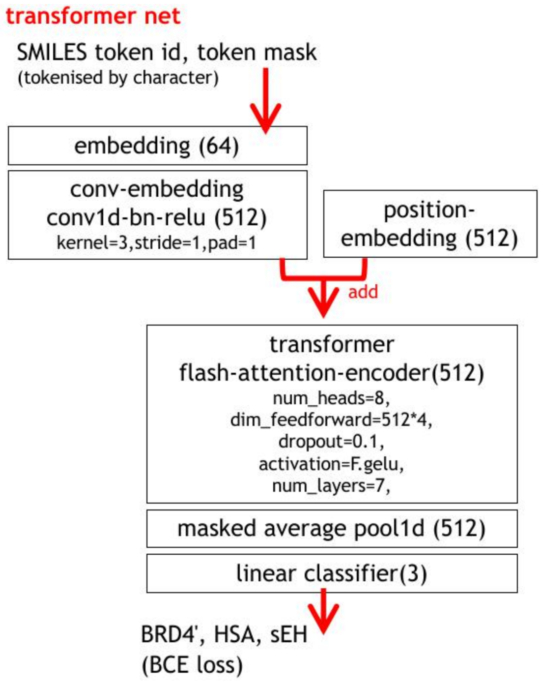
Background

- ◆ Contract computer vision and deep learning algorithm engineer.
  - identify fracture in x-ray images
  - implement visual slam for robotic navigation
  - finetune LLM models

- ◆ Familiar with deep learning and build deep models in my work.
- ◆ Experiences previous Kaggle competitions. Sequence modeling in molecule and medicine applications:
  - Bristol-Myers Squibb – Molecular Translation
  - OpenVaccine: COVID-19 mRNA Vaccine Degradation Prediction

Summary



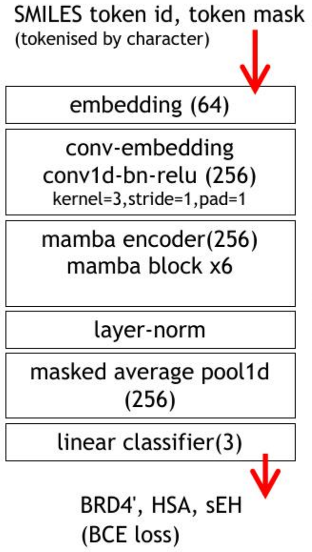


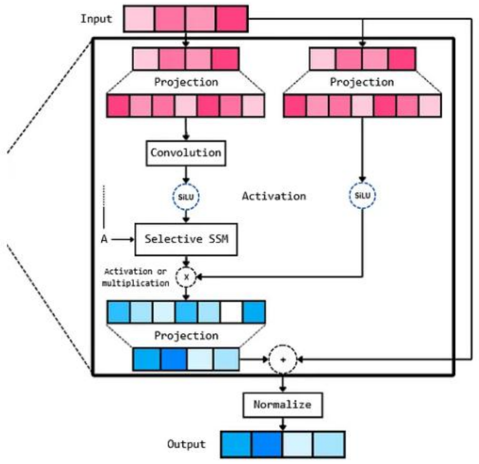
◆ To mitigate the effects of OOD (out-of-domain distribution), we use ensemble of different sequence models:

- 3-fold conv1d net
- 2-fold transformer net
- single-fold mamba (SSM, selective state machine)

◆ To handle 98 millions of training molecules efficiently, we use customized cuda kernel from open source flash attention[1] and mamba SSM[2] libraries

mamba net





Breakdown of a mamba block

◆ For training we have two Nvidia Ada A6000 / 48 GB Ampere GPUs. Time for training one fold with single GPU are:
- 7 hr for cnn1d,
- 28 hr for transformer,
- 36 hr for mamba.

			# keep score for the subset, set zero for others										
			all		keep-share #		keep-nonshare #		local CV (share)				
submission		**fold**	**private**	**public**	**private**	**public**	**private**	**public**	**fold0**	**fold1**	**fold2**	**fold3**	**fold4**
[1]+[2]+[3]	**final-3fold-tx2a-mamba-fix.submit.csv**		0.28557	0.46439	0.22761	0.34985	0.08320	0.10362	0.65976	0.66044		0.67601	
[1]	**final-3fold-cnn1d.submit.csv**	fold0,1,3	0.26615	0.41103	0.22693	0.34946	0.05424	0.08504					
[2]	**final-2fold-transfomer.submit.csv**	fold2,4	0.31236	0.42288	0.22613	0.34753	0.10124	0.09881			0.64041		0.64589
[3]	**final-1fold-mamba.submit.csv**	fold0	0.23905	0.43221	0.22404	0.34484	0.03002	0.11083	0.65559				
late submission													
[2a]	final-2fold-cnn1d.submit.csv	fold2,4	0.26403	0.40879	0.22581	0.34814	0.05324	0.08411			0.63988		0.64613
[2b]	final-2fold-mamba.submit.csv	fold2,4	0.26361	0.41035	0.22712	0.34668	0.05150	0.08713			0.63848		0.64400

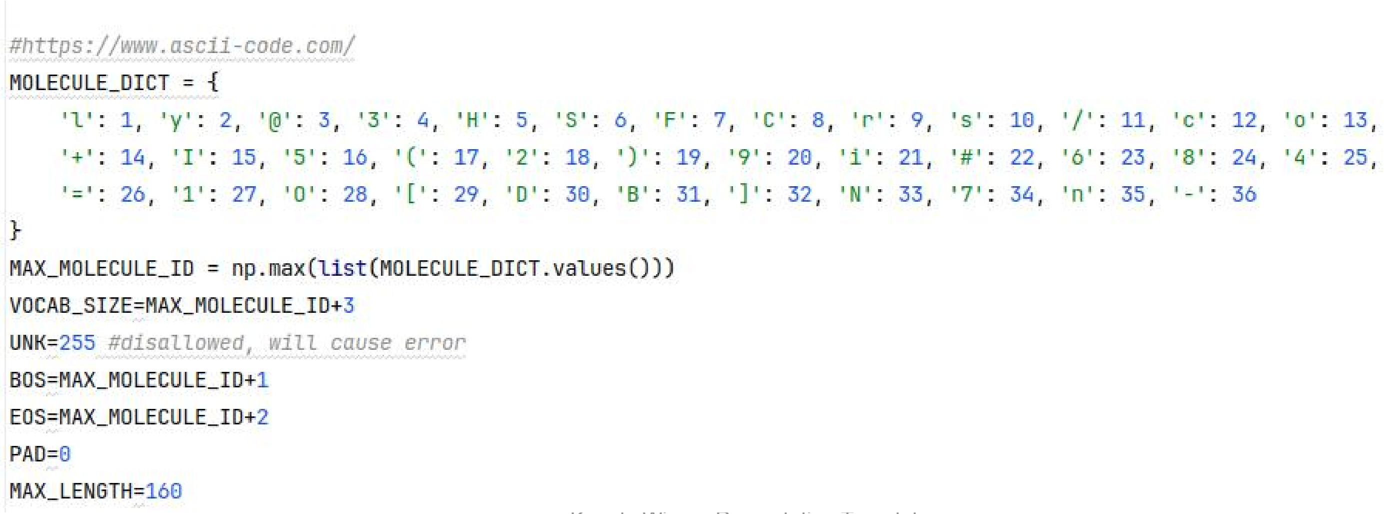
Features Selection/ Engineering

Features Selection/Engineering

◆ **Tokenization** :

We tried several methods like character based, sentence piece, byte-pair-encoding (BPE), atom/smiles notation aware to break the SMILES strings. Surprisingly, the **simplest character based tokenization performs the best** across different net architecture

Add a conv1d layer of kernel size=3, stride=1 to learned combinations of consecutive tokens (bi-grams, tri-grams) before passing them into cnn1d, transformer or mamba encoder.



```
#https://www.ascii-code.com/
MOLECULE_DICT = {
    'l': 1, 'y': 2, '@': 3, '3': 4, 'H': 5, 'S': 6, 'F': 7, 'C': 8, 'r': 9, 's': 10, '/': 11, 'c': 12, 'o': 13,
    '+': 14, 'I': 15, '5': 16, '(': 17, '2': 18, ')': 19, '9': 20, 'i': 21, '#': 22, '6': 23, '8': 24, '4': 25,
    '=': 26, '1': 27, 'O': 28, '[': 29, 'D': 30, 'B': 31, ']': 32, 'N': 33, '7': 34, 'n': 35, '-': 36
}
MAX_MOLECULE_ID = np.max(list(MOLECULE_DICT.values()))
VOCAB_SIZE=MAX_MOLECULE_ID+3
UNK=255 #disallowed, will cause error
BOS=MAX_MOLECULE_ID+1
EOS=MAX_MOLECULE_ID+2
PAD=0
MAX_LENGTH=160
```

Features Selection/Engineering

◆ **Batch normalization**:
cnn1d model performance is sensitive to batch normalization. Different feature values for:
- in-distribution and out-distribution samples.
- +ve and -ve samples due to class imbalance (less than 1% +ve).

To alleviate the problem, we use high eps=5e-3 and low momentum=0.2

Training Methods

Training Methods

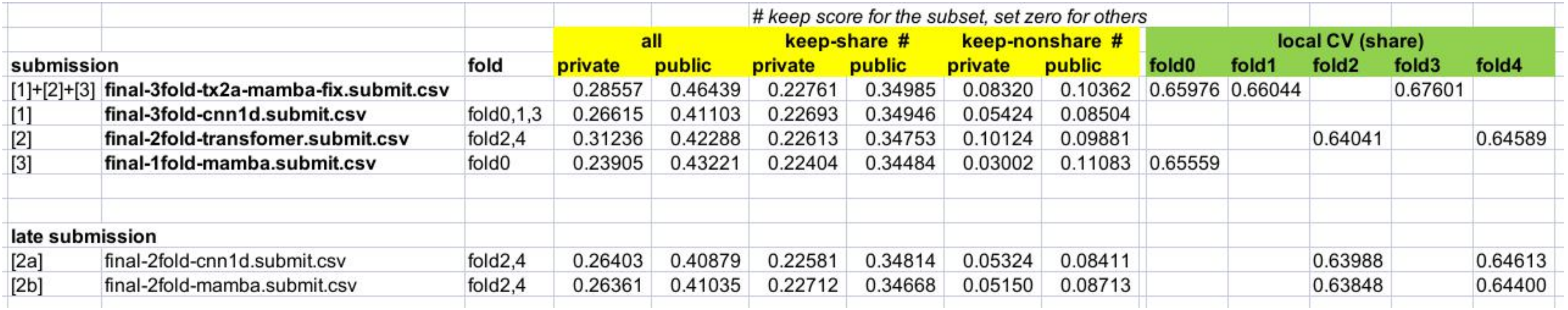
- binary cross entropy loss with ADAM optimizer.
  - step learning rate of 1e-3,1e-4,1e-5 for 6-12 epochs.
  - large batch size of 2000,2500,5000

Interesting, the best way to handle class imbalance is to do nothing (no up-sampling or under-sampling of the class). Maybe because of the large batch size we used.

Important and Interesting Findings

Important and Interesting Findings

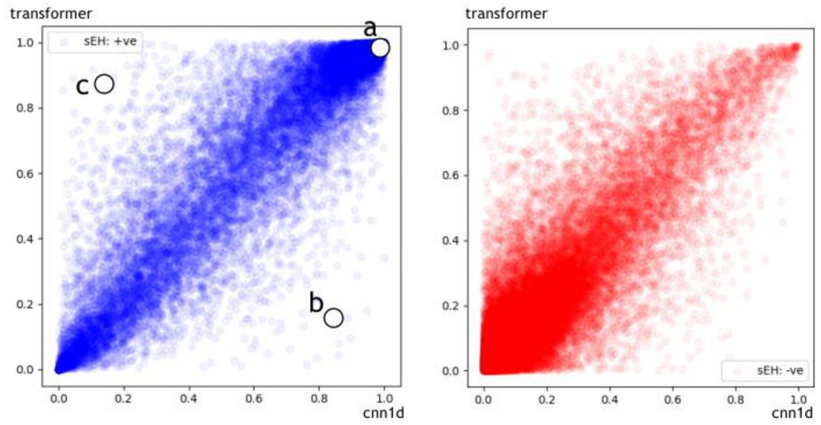
◆ Transformer is the most robust



			# keep score for the subset, set zero for others										
			all		keep-share #		keep-nonshare #		local CV (share)				
submission		fold	private	public	private	public	private	public	fold0	fold1	fold2	fold3	fold4
[1]+[2]+[3]	**final-3fold-tx2a-mamba-fix.submit.csv**		0.28557	0.46439	0.22761	0.34985	0.08320	0.10362	0.65976	0.66044		0.67601	
[1]	**final-3fold-cnn1d.submit.csv**	fold0,1,3	0.26615	0.41103	0.22693	0.34946	0.05424	0.08504					
[2]	**final-2fold-transfomer.submit.csv**	fold2,4	0.31236	0.42288	0.22613	0.34753	0.10124	0.09881			0.64041		0.64589
[3]	**final-1fold-mamba.submit.csv**	fold0	0.23905	0.43221	0.22404	0.34484	0.03002	0.11083	0.65559				
late submission													
[2a]	final-2fold-cnn1d.submit.csv	fold2,4	0.26403	0.40879	0.22581	0.34814	0.05324	0.08411			0.63988		0.64613
[2b]	final-2fold-mamba.submit.csv	fold2,4	0.26361	0.41035	0.22712	0.34668	0.05150	0.08713			0.63848		0.64400

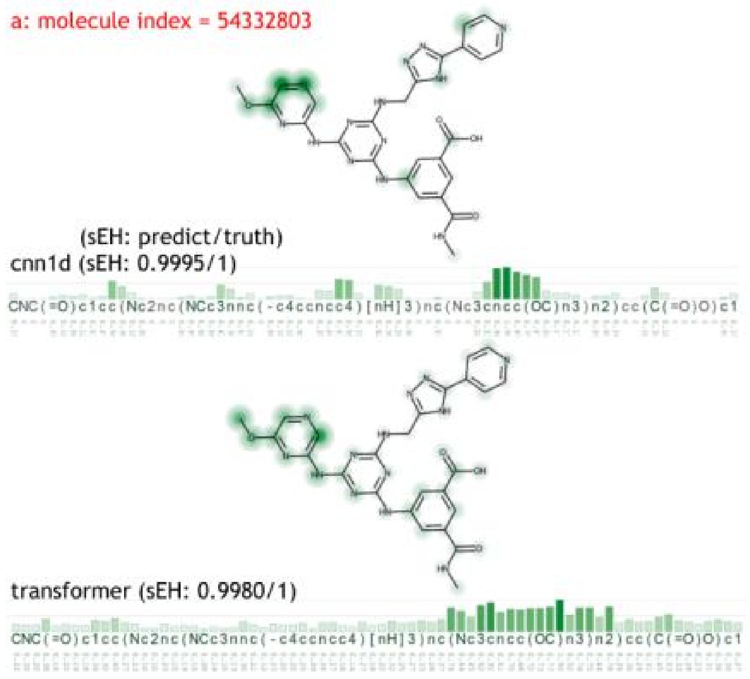
◆ correlation of cnn1d and transformer predictions

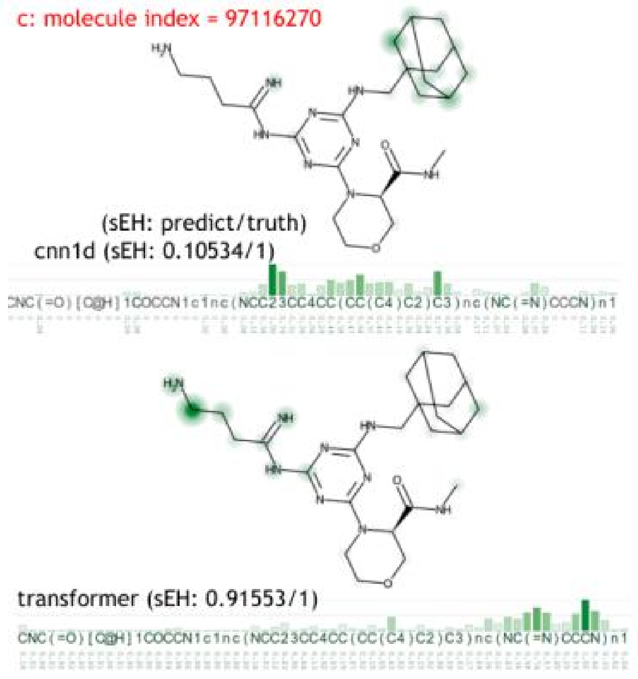
validation set: fold2 (4 million molecules)



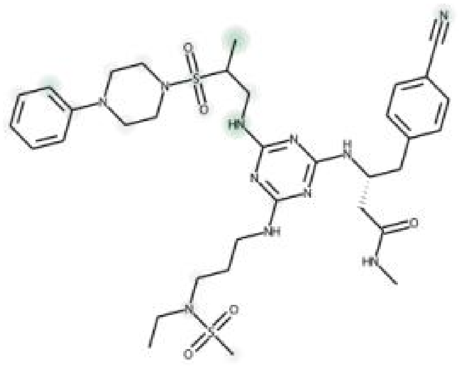
Important and Interesting Findings

◆ Generate the net prediction heatmap using GradCAM[3] and visualize it with XSMILES[4]. cnn1d has local activations, whereas transformer has more global ones.

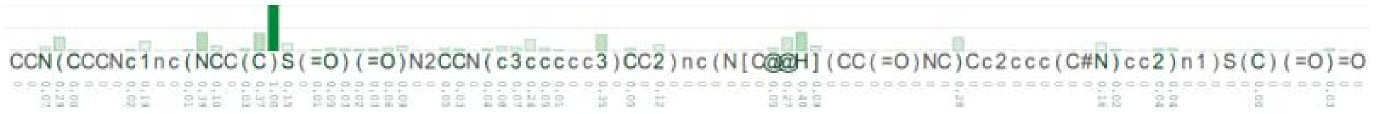

a: molecule index = 54332803
(sEH: predict/truth)
cnn1d (sEH: 0.9995/1)
CNC(=O)c1cc(Nc2nc(NCc3nnc(-c4ccncc4)[nH]3)nc(Nc3cncc(OC)n3)n2)cc(C(=O)O)c1
transformer (sEH: 0.9980/1)
CNC(=O)c1cc(Nc2nc(NCc3nnc(-c4ccncc4)[nH]3)nc(Nc3cncc(OC)n3)n2)cc(C(=O)O)c1


c: molecule index = 97116270
(sEH: predict/truth)
cnn1d (sEH: 0.10534/1)
CNC(=O)[C@H]1COCCN1c1nc(NCC23CC4CC(CC(C4)C2)C3)nc(NC(=N)CCCN)n1
transformer (sEH: 0.91553/1)
CNC(=O)[C@H]1COCCN1c1nc(NCC23CC4CC(CC(C4)C2)C3)nc(NC(=N)CCCN)n1

b: molecule index = 21821828



(sEH: predict/truth)
cnn1d (sEH: 0.9468/1)

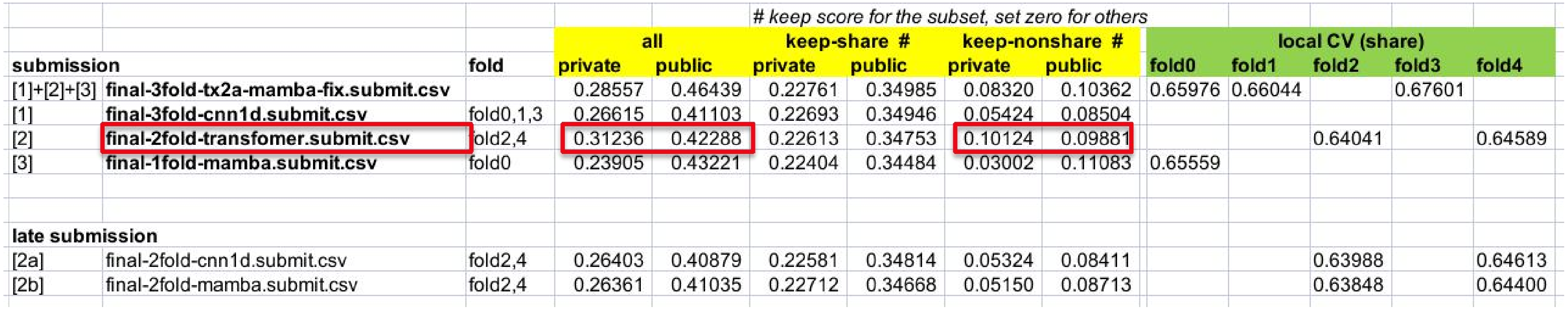


transformer (sEH: 0.1779/1)
heatmap is all zeros

Simple Model

Simple Model

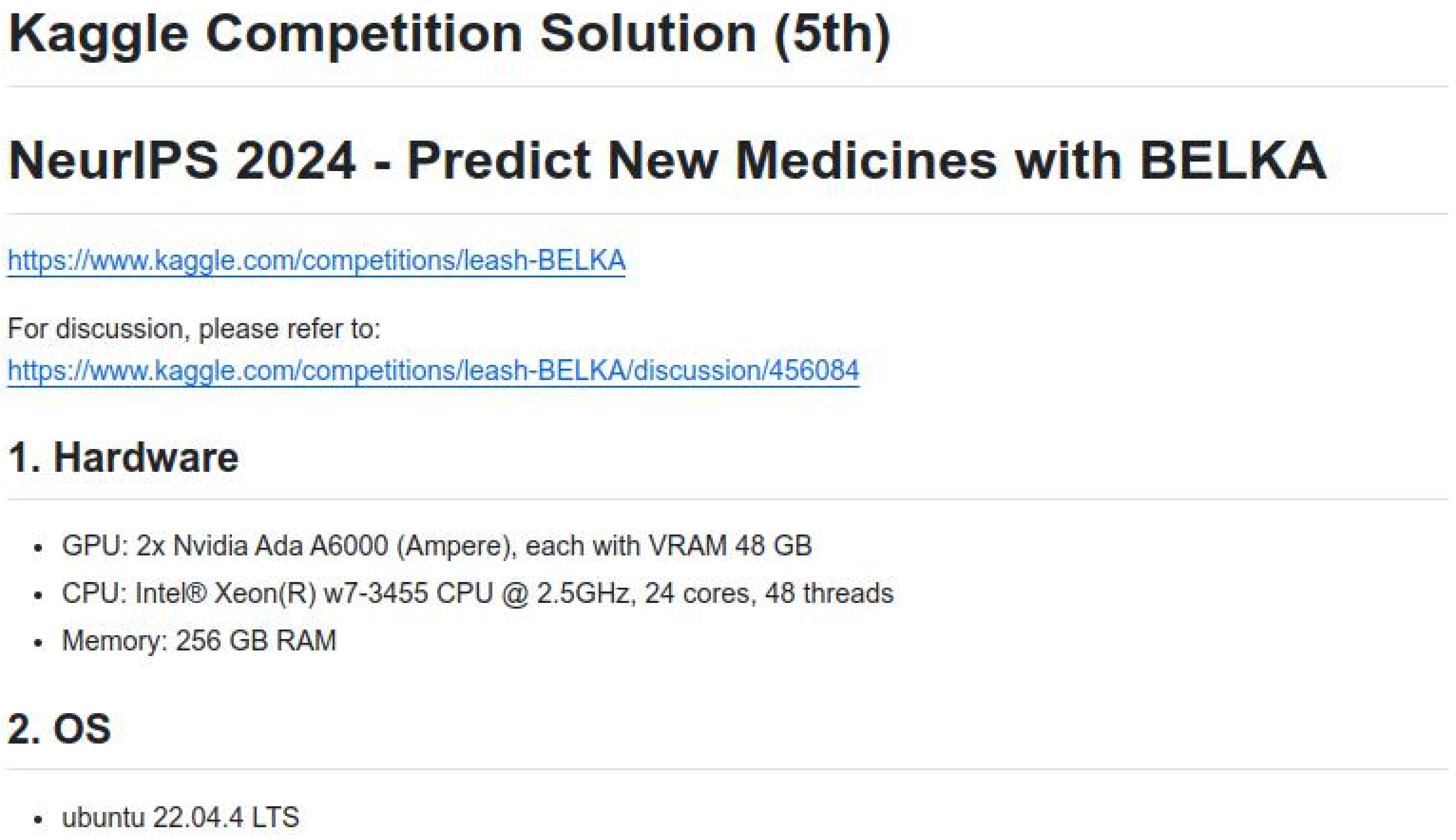
- Just transformer alone is good enough!



			# keep score for the subset, set zero for others										
			all		keep-share #		keep-nonshare #		local CV (share)				
submission		fold	private	public	private	public	private	public	fold0	fold1	fold2	fold3	fold4
[1]+[2]+[3]	final-3fold-tx2a-mamba-fix.submit.csv		0.28557	0.46439	0.22761	0.34985	0.08320	0.10362	0.65976	0.66044		0.67601	
[1]	final-3fold-cnn1d.submit.csv	fold0,1,3	0.26615	0.41103	0.22693	0.34946	0.05424	0.08504					
[2]	final-2fold-transfomer.submit.csv	fold2,4	0.31236	0.42288	0.22613	0.34753	0.10124	0.09881			0.64041		0.64589
[3]	final-1fold-mamba.submit.csv	fold0	0.23905	0.43221	0.22404	0.34484	0.03002	0.11083	0.65559				
late submission													
[2a]	final-2fold-cnn1d.submit.csv	fold2,4	0.26403	0.40879	0.22581	0.34814	0.05324	0.08411			0.63988		0.64613
[2b]	final-2fold-mamba.submit.csv	fold2,4	0.26361	0.41035	0.22712	0.34668	0.05150	0.08713			0.63848		0.64400

Code Review

https://github.com/hengck23/solution-leash-BELKA



Kaggle Competition Solution (5th)

NeurIPS 2024 - Predict New Medicines with BELKA

https://www.kaggle.com/competitions/leash-BELKA

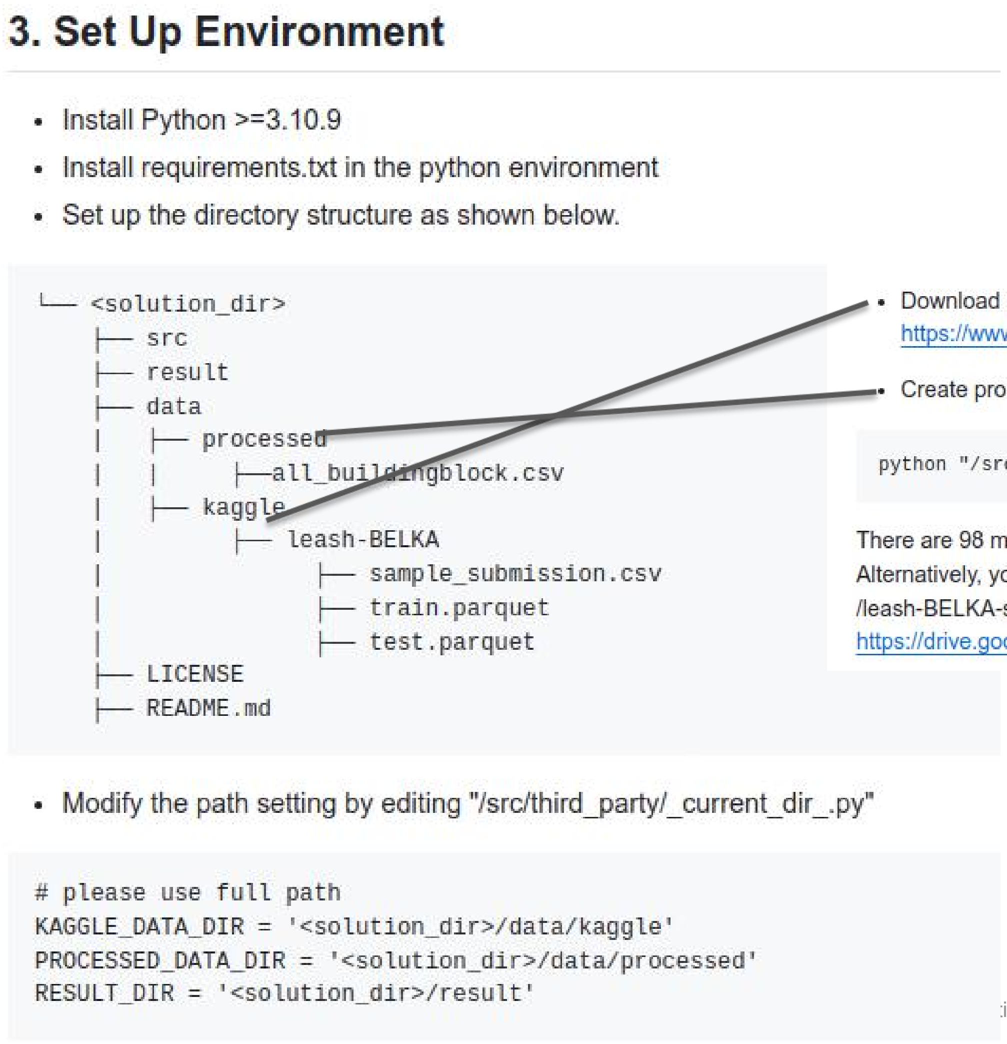
For discussion, please refer to:
https://www.kaggle.com/competitions/leash-BELKA/discussion/456084

1. Hardware

- GPU: 2x Nvidia Ada A6000 (Ampere), each with VRAM 48 GB
- CPU: Intel® Xeon(R) w7-3455 CPU @ 2.5GHz, 24 cores, 48 threads
- Memory: 256 GB RAM

2. OS

- ubuntu 22.04.4 LTS



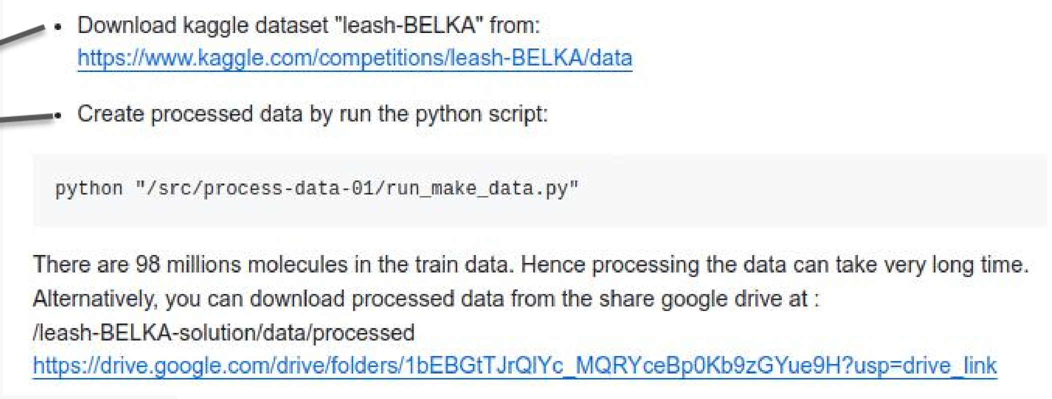
3. Set Up Environment

- Install Python >=3.10.9
- Install requirements.txt in the python environment
- Set up the directory structure as shown below.

```
└── <solution_dir>
    ├── src
    ├── result
    ├── data
    |   ├── processed
    |   |      ├──all_buildingblock.csv
    |   ├── kaggle
    |          ├── leash-BELKA
    |                 ├── sample_submission.csv
    |                 ├── train.parquet
    |                 ├── test.parquet
    ├── LICENSE
    ├── README.md
```

- Modify the path setting by editing "/src/third_party/_current_dir_.py"

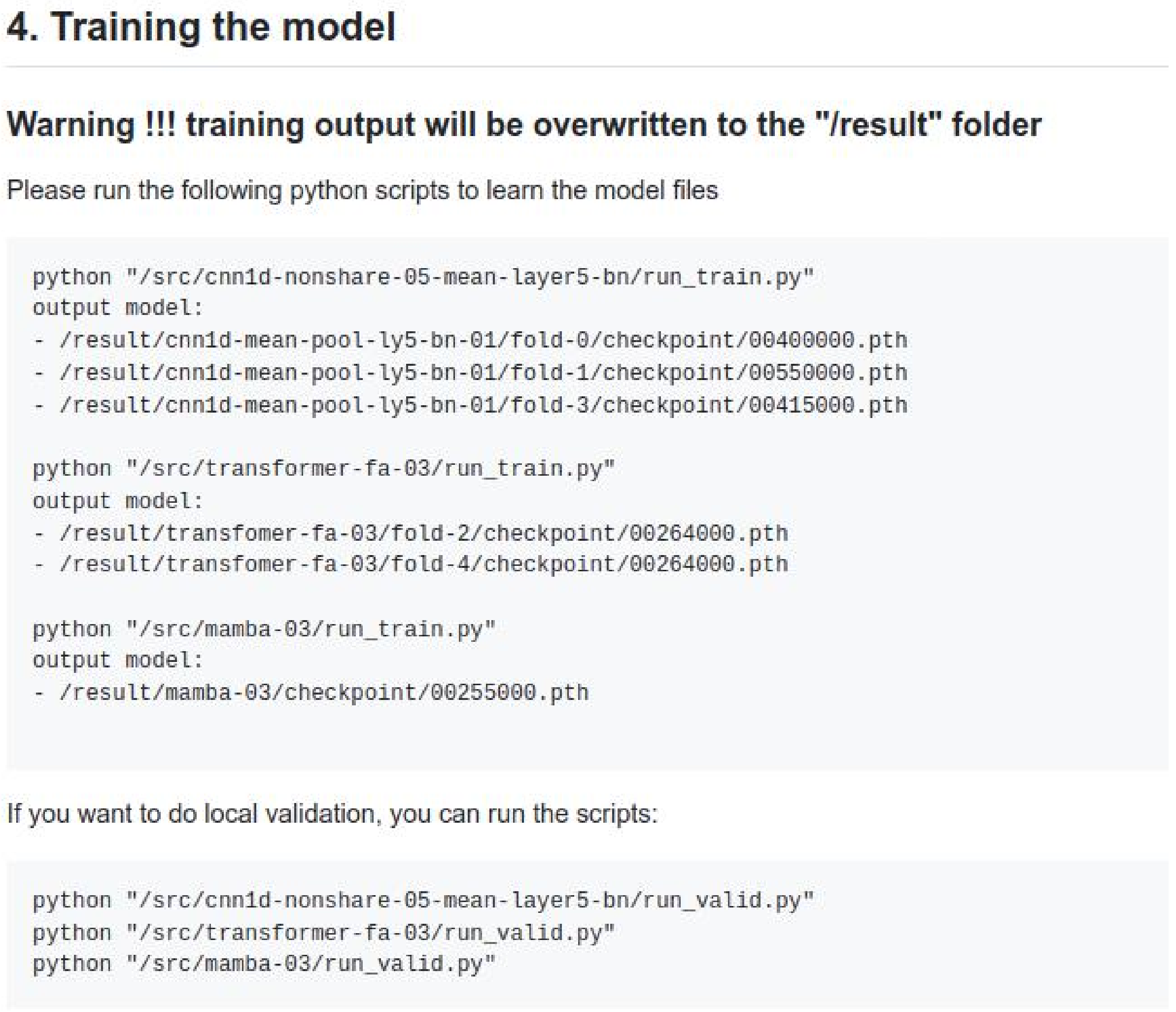
```
# please use full path
KAGGLE_DATA_DIR = '<solution_dir>/data/kaggle'
PROCESSED_DATA_DIR = '<solution_dir>/data/processed'
RESULT_DIR = '<solution_dir>/result'
```



- Download kaggle dataset "leash-BELKA" from: https://www.kaggle.com/competitions/leash-BELKA/data
- Create processed data by run the python script:

```
python "/src/process-data-01/run_make_data.py"
```

There are 98 millions molecules in the train data. Hence processing the data can take very long time. Alternatively, you can download processed data from the share google drive at : /leash-BELKA-solution/data/processed https://drive.google.com/drive/folders/1bEBGtTJrQlYc_MQRYceBp0Kb9zGYue9H?usp=drive_link



4. Training the model

Warning !!! training output will be overwritten to the "/result" folder

Please run the following python scripts to learn the model files

```
python "/src/cnn1d-nonshare-05-mean-layer5-bn/run_train.py"
output model:
- /result/cnn1d-mean-pool-ly5-bn-01/fold-0/checkpoint/00400000.pth
- /result/cnn1d-mean-pool-ly5-bn-01/fold-1/checkpoint/00550000.pth
- /result/cnn1d-mean-pool-ly5-bn-01/fold-3/checkpoint/00415000.pth

python "/src/transformer-fa-03/run_train.py"
output model:
- /result/transfomer-fa-03/fold-2/checkpoint/00264000.pth
- /result/transfomer-fa-03/fold-4/checkpoint/00264000.pth

python "/src/mamba-03/run_train.py"
output model:
- /result/mamba-03/checkpoint/00255000.pth
```

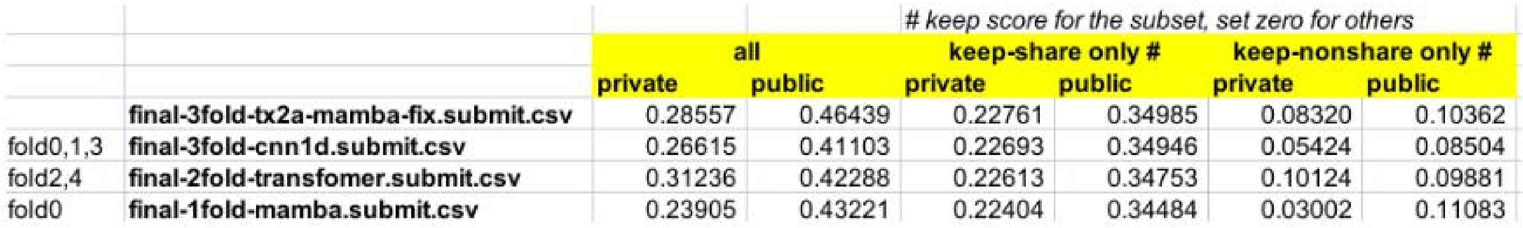
If you want to do local validation, you can run the scripts:

```
python "/src/cnn1d-nonshare-05-mean-layer5-bn/run_valid.py"
python "/src/transformer-fa-03/run_valid.py"
python "/src/mamba-03/run_valid.py"
```

5. Submission csv

Please run the following scripts:

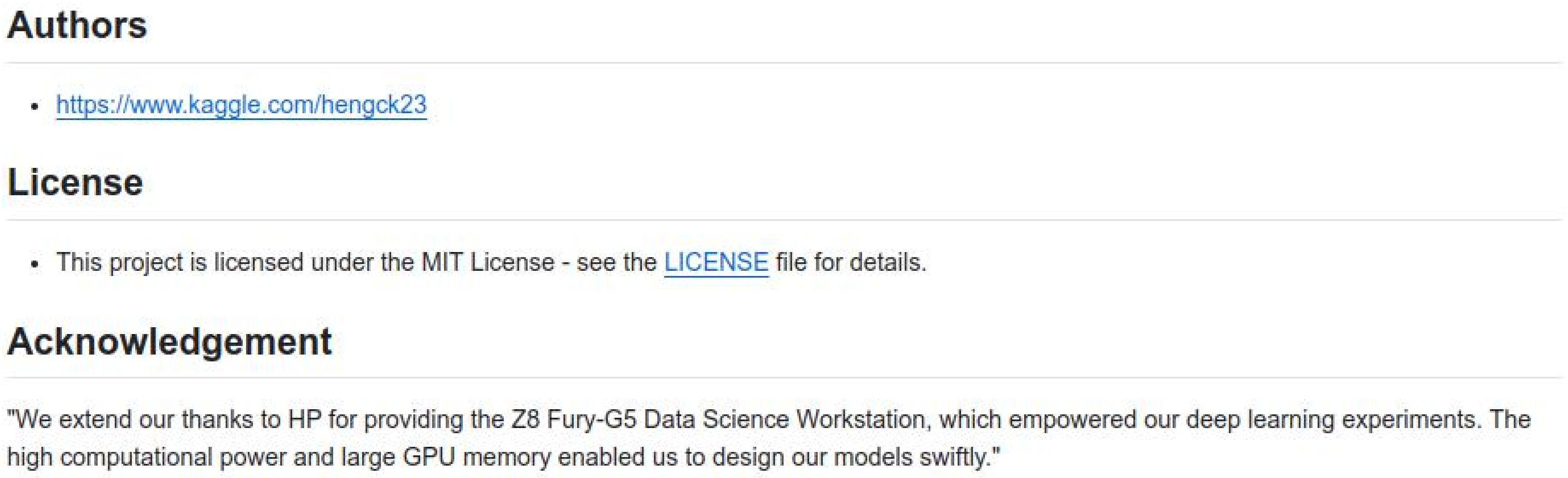
```
python "/src/cnn1d-nonshare-05-mean-layer5-bn/run_submit.py"
python "/src/transformer-fa-03/run_submit.py"
python "/src/mamba-03/run_submit.py"
python "/src/run_ensemble.py"
output file:
- /result/final-3fold-tx2a-mamba-fix.submit.csv
```



				# keep score for the subset, set zero for others			
		all		keep-share only #		keep-nonshare only #	
		private	public	private	public	private	public
	final-3fold-tx2a-mamba-fix.submit.csv	0.28557	0.46439	0.22761	0.34985	0.08320	0.10362
fold0,1,3	**final-3fold-cnn1d.submit.csv**	0.26615	0.41103	0.22693	0.34946	0.05424	0.08504
fold2,4	**final-2fold-transfomer.submit.csv**	0.31236	0.42288	0.22613	0.34753	0.10124	0.09881
fold0	**final-1fold-mamba.submit.csv**	0.23905	0.43221	0.22404	0.34484	0.03002	0.11083

6. Reference trained models and validation results

- Reference results can also be found in the share google drive at :
  /leash-BELKA-solution/result
  https://drive.google.com/drive/folders/1bEBGtTJrQlYc_MQRYceBp0Kb9zGYue9H?usp=drive_link
- It includes the weight files, train/validation logs.



Authors

- https://www.kaggle.com/hengck23

License

Acknowledgement

"We extend our thanks to HP for providing the Z8 Fury-G5 Data Science Workstation, which empowered our deep learning experiments. The high computational power and large GPU memory enabled us to design our models swiftly."

Question and Answer

kaggle