Assignment 2 - Kaggle Part
Antibiotic Resistance Genes (ARGs) Prediction

AIST 4010: Foundation of Applied Deep Learning (Spring 2022)

DUE: **11:59PM (HKT), Mar. 18, 2022**

1 Introduction

The spread of antibiotic resistance has become one of the most urgent threats to global health, which is estimated to cause 700,000 deaths each year globally. Its surrogates, antibiotic resistance genes (ARGs), are highly transmittable between food, water, animal, and human to mitigate the efficacy of antibiotics. Accurately identifying ARGs is thus an indispensable step to understanding the ecology, and transmission of ARGs between environmental and human-associated reservoirs.

In this assignment, you will use CNN or LSTM or even attention mechanism to design an end-to-end system for ARG prediction. In this task, your system will be given raw sequence protein sequences (translated from genes) as input. First, your system needs to classify the input into ARG or non-ARG. Then if the input is a non-ARG, your system needs to output the class **'nonarg'**. If the input is an ARG, your system needs go futher to predict resistant antibiotic type. Here we have 14 antibiotic families including **'aminoglycoside', 'macrolide-lincosamide-streptogramin', 'polymyxin', 'fosfomycin', 'trimethoprim', 'bacitracin', 'quinolone', 'multidrug', 'chloramphenicol','tetracycline', 'rifampin', 'beta_lactam', 'sulfonamide', 'glycopeptide'**. The system should decide which family the predicted ARG resistant to.

You will train your model on a dataset containing thousands of sequences labeled by non-ARG or ARG with the corresponding families. You will learn more about sequence processing, sequence encoding, and, of course, convolutional or other neural network layers. This assignment is quite different from assignment 1. You will develop skills of processing sequences with deep learning models.

- Goal: Given a raw protein sequence, predict the non-ARG or ARG family class.
- Kaggle: `https://www.kaggle.com/c/aist4010-spring2022-a2`

2 ARG Prediction

2.1 Protein Sequence Representation

We all know images are represented by many digit values for every pixel. Then how will we represent protein sequences? Protein sequence is typically notated as a string of letters, listing the amino acids starting at the amino-terminal(N) end through to the carboxyl-terminal(C) end. Either a three letter code or single letter code can be used to represent the 20 naturally occurring amino acids. Table 1 shows the 20 natural amino acid notation. For example, a protein sequence can be represented as 'MKLIEIEKLNKYFNTAIIGAS'.

Table 1: 20 natural amino acid notation

Amino Acid	3-Letter	1-Letter
Alanine	Ala	A
Arginine	Arg	R
Asparagine	Asn	N
Aspartic acid	Asp	D
Cysteine	Cys	C
Glutamic acid	Glu	E
Glutamine	Gln	Q
Glycine	Gly	G
Histidine	His	H
Isoleucine	Ile	I
Leucine	Leu	L
Lysine	Lys	K
Methionine	Met	M
Phenylalanine	Phe	F
Proline	Pro	P
Serine	Ser	S
Threonine	Thr	T
Tryptophan	Trp	W
Tyrosine	Tyr	Y
Valine	Val	V

2.2 Protein Sequence Encoding

Image pixel values are natural digits that can be passed to our algorithms to process. However, for the letters of protein sequences, we couldn't directly deal with them. So the first problem you need to solve is how to encode the protein sequences. Actually, there are several ways to achieve it.

Here, **One-hot encoding** is the simplest but very effective solution. By One-hot encoding, the protein sequence can be encoded into a $20 \times L$ matrix. Each row corresponds to the presence of 20 standard amino acid (AAs) while each column is a spot on a protein sequence.

For one-hot encoding, you may need to think about two questions:

- What if we meet a rare amino acid besides the 20 standard AAs?
- How to deal with sequences of different lengths?

Besides, you can also apply some **pre-trained protein language models** to represent the protein sequences like ESM[1] and ProtTrans[2]. These models are trained on large protein datasets. Unlike 'VGGFace' targeting exactly the same task as A1, these models are for general purpose. But they could be applied to various downstream tasks including contact prediction, structure predictions, interaction prediction etc., and they lead very amazing results. So, you can try them to see if you could benefit from these pre-trained models.

2.3 Multi-class Classification

This problem may seem fancy, but the ARG prediction is still a multi-class classification: the input to your system is a protein sequence and your model needs to predict the non-ARG or ARG class.

[1] https://github.com/facebookresearch/esm

[2] https://github.com/agemagician/ProtTrans

However, noted that the number of non-ARG sequences is a lot larger than that of any single ARG class. Therefore, this dataset is actually an **imbalanced** dataset. You need to think about how to deal with this problem.

The task could be divided to three parts:

- Loading sequence data and labels from raw files.
- Transforming the data to appropriate representations.
- Training a deep learning model for classification.

3 Dataset

The data for the assignment can be downloaded from the Kaggle competition link[3]. The dataset contains thousands of protein sequences with labels.

3.1 File Structure

The structure of the dataset is as follows:

- `train.fasta`: You are supposed to use train data set to train your model for the task.
- `val.fasta`: You are supposed to use val data to validate the classification F-score.
- `test.fasta`: You are supposed to assign classes for images in test data and submit your result.
- `sample-submission.csv`: This is a sample submission file for this competition.

3.2 Loading Sequence Data - Bio.SeqIO

All the protein sequences are stored as the 'FASTA' format. In bioinformatics, the 'FASTA' format is a text-based format for representing either nucleotide sequences or amino acid (protein) sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences.

For one sequence, usually it includes two parts: the description and the sequences as shown in figure 1. First line beginning with **'>'** symbol is the description while the second line is the protein sequence information. You can easily deal with this type of files by **'Bio.SeqIO'** library[4].

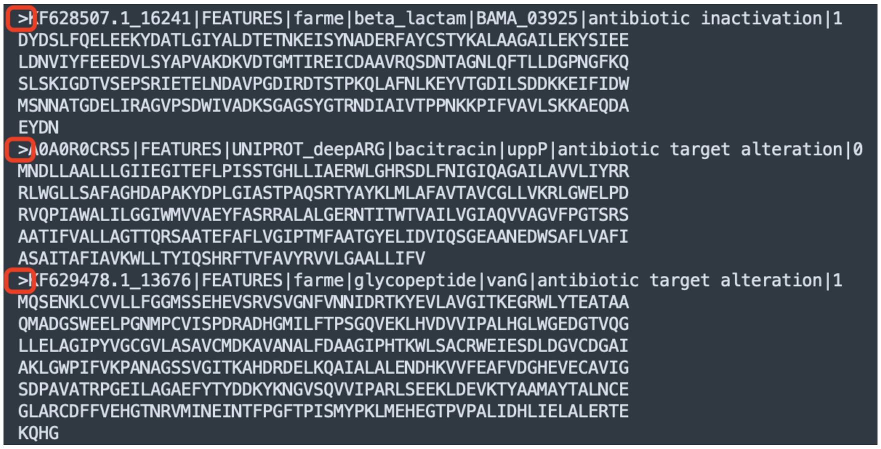
3.3 Loading Data Labels

For an ARG sequence as shown in figure 2, the second part of the description is **'FEATURES'**, and the forth part is the ARG class. In this example, the ARG class is **'quinolone'**.

For a non-ARG sequence as shown in figure 3, the description starts with **'sp'** and doesn't have **'FEATURES'** or ARG classes.

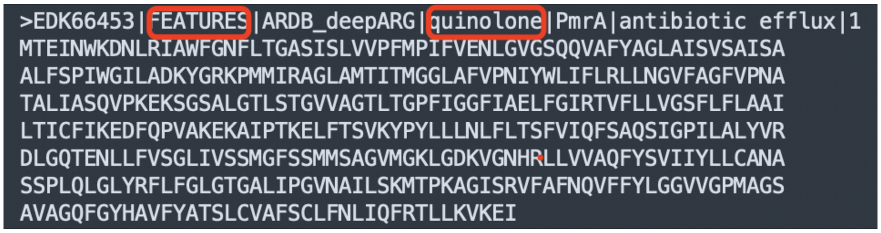
[3] `https://www.kaggle.com/c/aist4010-spring2022-a2/data`

[4] `https://biopython.org/wiki/SeqIO`



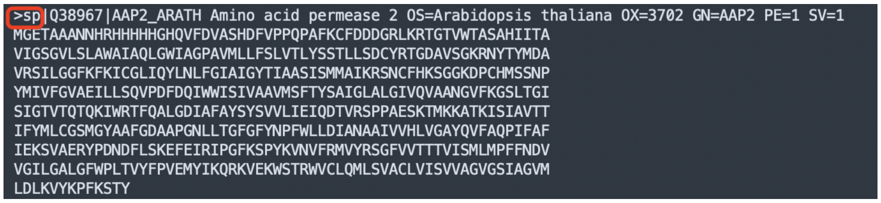
```
>KF628507.1_16241|FEATURES|farme|beta_lactam|BAMA_03925|antibiotic inactivation|1
DYDSLFQELEEKYDATLGIYALDTETNKEISYNADERFAYCSTYKALAAGAILEKYSIEE
LDNVIYFEEEDVLSYAPVAKDKVDTGMTIREICDAAVRQSDNTAGNLQFTLLDGPNGFKQ
SLSKIGDTVSEPSRIETELNDAVPGDIRDTSTPKQLAFNLKEYVTGDILSDDKKEIFIDW
MSNNATGDELIRAGVPSDWIVADKSGAGSYGTRNDIAIVTPPNKKPIFVAVLSKKAEQDA
EYDN
>A0A0R0CRS5|FEATURES|UNIPROT_deepARG|bacitracin|uppP|antibiotic target alteration|0
MNDLLAALLLGIIEGITEFLPISSTGHLLIAERWLGHRSDLFNIGIQAGAILAVVLIYRR
RLWGLLSAFAGHDAPAKYDPLGIASTPAQSRTYAYKLMLAFAVTAVCGLLVKRLGWELPD
RVQPIAWALILGGIWMVVAEYFASRRALALGERNTITWTVAILVGIAQVVAGVFPGTSRS
AATIFVALLAGTTQRSAATEFAFLVGIPTMFAATGYELIDVIQSGEAANEDWSAFLVAFI
ASAITAFIAVKWLLTYIQSHRFTVFAVYRVVLGAALLIFV
>KF629478.1_13676|FEATURES|farme|glycopeptide|vanG|antibiotic target alteration|1
MQSENKLCVVLLFGGMSSEHEVSRVSVGNFVNNIDRTKYEVLAVGITKEGRWLYTEATAA
QMADGSWEELPGNMPCVISPDRADHGMILFTPSGQVEKLHVDVVIPALHGLWGEDGTVQG
LLELAGIPYVGCGVLASAVCMDKAVANALFDAAGIPHTKWLSACRWEIESDLDGVCDGAI
AKLGWPIFVKPANAGSSVGITKAHDRDELKQAIALALENDHKVVFEAFVDGHEVECAVIG
SDPAVATRPGEILAGAEFYTYDDKYKNGVSQVVIPARLSEEKLDEVKTYAAMAYTALNCE
GLARCDFFVEHGTNRVMINEINTFPGFTPISMYPKLMEHEGTPVPALIDHLIELALERTE
KQHG
```

Figure 1: A ‘FASTA’ files example



```
>EDK66453|FEATURES|ARDB_deepARG|quinolone|PmrA|antibiotic efflux|1
MTEINWKDNLRIAWFGNFLTGASISLVVPFMPIFVENLGVGSQQVAFYAGLAISVSAISA
ALFSPIWGILADKYGRKPMMIRAGLAMTITMGGLAFVPNIYWLIFLRLLNGVFAGFVPNA
TALIASQVPKEKSGSALGTLSTGVVAGTLTGPFIGGFIAELFGIRTVFLLVGSFLFLAAI
LTICFIKEDFQPVAKEKAIPTKELFTSVKYPYLLLNLFLTSFVIQFSAQSIGPILALYVR
DLGQTENLLFVSGLIVSSMGFSSMMSAGVMGKLGDKVGNHRLLVVAQFYSVIIYLLCANA
SSPLQLGLYRFLFGLGTGALIPGVNAILSKMTPKAGISRVFAFNQVFFYLGGVVGPMAGS
AVAGQFGYHAVFYATSLCVAFSCLFNLIQFRTLLKVKEI
```

Figure 2: An ARG sequence example



```
>sp|Q38967|AAP2_ARATH Amino acid permease 2 OS=Arabidopsis thaliana OX=3702 GN=AAP2 PE=1 SV=1
MGETAAANNHRHHHHHGHQVFDVASHDFVPPQPAFKCFDDDGRLKRTGTVWTASAHIITA
VIGSGVLSLAWAIAQLGWIAGPAVMLLFSLVTLYSSTLLSDCYRTGDAVSGKRNYTYMDA
VRSILGGFKFKICGLIQYLNLFGIAIGYTIAASISMMAIKRSNCFHKSGGKDPCHMSSNP
YMIVFGVAEILLSQVPDFDQIWWISIVAAVMSFTYSAIGLALGIVQVAANGVFKGSLTGI
SIGTVTQTQKIWRTFQALGDIAFAYSYSVVLIEIQDTVRSPPAESKTMKKATKISIAVTT
IFYMLCGSMGYAAFGDAAPGNLLTGFGFYNPFWLLDIANAAIVVHLVGAYQVFAQPIFAF
IEKSVAERYPDNDFLSKEFEIRIPGFKSPYKVNVFRMVYRSGFVVTTTVISMLMPFFNDV
VGILGALGFWPLTVYFPVEMYIKQRKVEKWSTRWVCLQMLSVACLVISVVAGVGSIAGVM
LDLKVYKPFKSTY
```

Figure 3: A non-ARG sequence example

4 System Evaluation

4.1 Label Mapping

For evaluation, please mapping your predicted ARG classes as this dict.
arg_dict =
{‘aminoglycoside’: 0, ‘macrolide-lincosamide-streptogramin’: 1, ‘polymyxin’: 2, ’fosfomycin’: 3, ‘trimethoprim’: 4, ‘bacitracin’: 5, ‘quinolone’: 6, ‘multidrug’: 7, ‘chloramphenicol’: 8, ‘tetracycline’: 9, ‘rifampin’: 10, ‘beta_lactam’: 11, ‘sulfonamide’: 12, ‘glycopeptide’: 13}
If the predicted result is non-ARG, please set the label to be **{‘nonarg’: 14}**.

You may check the `sample-submission.csv` for your reference before submission.

4.2 Evaluation Metric

The evaluation metric is macro F-score. To calculate macro F-score, first we need to calculate macro average of Precision and Recall. Then we apply the F-score equation to calculate it.

$$\begin{aligned} \text{PrecisionMacroAvg} &= \frac{(Prec_1 + Prec_2 + \ldots + Prec_n)}{n} \\ \text{RecallMacroAvg} &= \frac{(Rec_1 + Rec_2 + \ldots + Rec_n)}{n} \\ \text{F1} &= 2 \cdot \frac{\text{PrecisionMacroAvg} \cdot \text{RecallMacroAvg}}{\text{PrecisionMacroAvg} + \text{RecallMacroAvg}} \end{aligned} \tag{1}$$

5 Submission

Following are the deliverables for this assignment:

- Kaggle submission. Please submit your results on Kaggle page with your nickname. Make sure your nickname appears on the public leaderboard. If you are better than the baseline, you can be above 60%. If you are better than the deep learning baseline, you can be above 80%.
- Blackboard submissions. Please pack all files in one '.zip' file named as **'nickname_real name_SID'** like 'lctest_Licheng ZONG_1155123456.zip'. Please name your files as **this format**, or you will get **mark deduction**.
  - A one page report describing your model architecture, hyper parameters, the epochs you trained and any other interesting details leading to your best result for the above competition. Please limit the report to **one page**.
  - **All** your source codes as the format of '.ipynb' or '.py' in **one folder**.

6 Conclusion

Nicely done! Here is the end of Assignment 2 (Kaggle Part), and the beginning of the sequence processing world. As always, feel free to ask on Piazza if you have any questions. We are always here to help.

Good luck and enjoy the challenge!

7 Reference

1. `https://en.wikipedia.org/wiki/Protein_primary_structure`
2. `https://en.wikipedia.org/wiki/FASTA_format`
3. `https://tomaxent.com/2018/04/27/Micro-and-Macro-average-of-Precision-Recall-and-F-Score/`
4. `https://biopython.org/wiki/SeqIO`
5. `https://github.com/facebookresearch/esm`
6. `https://github.com/agemagician/ProtTrans`
7. Junkang Wei, Siyuan Chen, Licheng Zong, Xin Gao, Yu Li, Protein–RNA interaction prediction with deep learning: structure matters, Briefings in Bioinformatics, Volume 23, Issue 1, January 2022