Predicting A Virus’s Infectivity Using Its Genomes

Members: Elias Xu, Mark Ionis, Seth Fenton, Jakob Weir
Mentor: Ryan Goggins

THE PREMISE:

- R0 value - indicates how easily a virus spreads, great measure of infectivity

- R0 values are hard to calculate, require massive logistically difficult experiments

- Everything in a virus coded for by genes

- Stored in the genome, AKA a long sequence of four letters ACG and T (Or U if an RNA Virus)

THE GOAL:

Build an ML model to predict transmission level based on a viruses genome

The Reasoning:

Viral genomes are incredibly easy to find as opposed to R0 values

Found 380 viruses with known genomes and transmission levels by cross referencing a viral genome databank and a paper containing said transmission levels

Virus Name	Genome	Estimated Transmission Level
Mamastrovirus 4	AAGAAGGAGGTTATCAAAGAGGAAAAGATCAAGAACAATGACATCC...	5
Indiana vesiculovirus	ACGAAGACAAACAAACCATTATTATCATTAAAAGGCTCAGGAGAAA...	2
Indiana vesiculovirus	ACGAAGACAAACAAACCATTATTACCATTAAAAGGCTCAGGAGAAA...	2
Human orthorubulavirus 4	ACCAAGGGGAGAAGAGATATGGATACTGATCTGGAAAATTAAAGGT...	5
Bundibugyo ebolavirus	CGGACACACAAAAAGAATGAAGGATTTTGAATCTTTATTGTGTGCG...	3

Tokenized the genomes into lists of numbers each correlated to ATCG or U

```
tensor([2., 2., 4., 3., 3., 3., 3., 1., 1., 1., 2., 4., 2., 4., 3., 1., 4., 1.,
        2., 3., 2., 1., 3., 2., 3., 4., 3., 4., 3., 4., 1., 2., 3., 1., 3., 3.,
        2., 1., 4., 2., 3., 2., 1., 4., 2., 1., 2., 3., 4., 3., 3., 4., 4., 3.,
        3., 4., 3., 1., 3., 3., 4., 4., 3., 3., 4., 1., 1., 4., 3., 1., 3., 1.,
        3., 1., 3., 1., 1., 2., 1., 1., 3., 2., 1., 1., 3., 4., 2., 3., 4., 3.,
        3., 4., 1., 1., 4., 1., 1., 4., 2., 3., 4., 1., 1., 4., 3., 4., 3., 3.,
        3., 4., 3., 2., 2., 1., 1., 1., 1., 3., 4., 3., 1., 3., 3., 2., 2., 3.,
        1., 2., 2., 1., 2., 3., 2., 1., 1., 4., 1., 2., 3., 1., 4., 1., 3., 3.,
        3., 4., 1., 4., 4., 1., 2., 1., 3., 2., 4., 1., 2., 1., 1., 2., 4., 1.,
        3., 3., 4., 1., 1., 3., 1., 1., 1., 3., 1., 4., 1., 3., 2., 4., 3., 2.,
        3., 4., 2., 1., 1., 2., 3., 4., 3., 4., 4., 1., 2., 1., 2., 2., 1., 3.,
        1., 3., 3., 2., 2., 1., 4., 4., 1., 3., 4., 3., 2., 1., 3., 3., 4., 3.,
        2., 4., 3., 3., 4., 1., 1., 1., 4., 2., 1., 3., 4., 4., 1., 4., 3., 4.,
        1., 4., 4., 1., 2., 3., 3., 3., 3., 3., 4., 2., 1., 3., 3., 1., 2., 1.,
```

Built a transformer model via PyTorch

Create our dataloader

```python
import torch
from torch.utils.data import Dataset

class viralDataset(Dataset):
    def __init__(self, sequences: list, r0_values: np.ndarray, tokenizer
        self.sequences = sequences
        self.r0_values = r0_values
        self.tokenizer = tokenizer

    def __len__(self):
        return self.r0_values.shape[0]

    def __getitem__(self, index):
        try:
            sequence = self.sequences[index].tolist()
            # sequence = self.tokenizer(sequence)
            # Ensure the sequence is a PyTorch tensor
            sequence = torch.tensor(sequence, dtype=torch.float32)
        except Exception as e:
            print(f"Error processing sequence at index {index}: {e}")
            raise
```

Positional Encoding

```python
rt math
rt torch
rt torch.nn as nn


s PositionalEncoding(nn.Module):

def __init__(self, d_model: int, max_len: int = 5000):
    super().__init__()
    self.dropout = nn.Dropout(p=0.1)

    position = torch.arange(max_len).unsqueeze(1)
    div_term = torch.exp(
        torch.arange(0, d_model, 2) * (-math.log(10000.0)
    )
    pe = torch.zeros(max_len, 1, d_model)
    pe[:, 0, 0::2] = torch.sin(position * div_term)
    pe[:, 0, 1::2] = torch.cos(position * div_term)
    self.register_buffer("pe", pe)

def forward(self, x):
    """
    Arguments:
        x: Tensor, shape ``[seq_len, batch_size, embedding
    """
    x = x + self.pe[: x.size(0), None, :]
    return self.dropout(x)
```

Create our train and test loop and begin training

```python
[39] # initialize our model
import torch
import torch.nn as nn

model = GenomeR0ValueModel(
    ntoken=len(vocabs), d_model=64, nhead=8, d_hid=256, nlayers=8
)
```

```python
import torch
from torch.utils.data import DataLoader
from tqdm import tqdm  # Progress bar (optional, but very helpful)


def train_model(
    model,
    train_loader,
    test_loader,
    criterion,
    optimizer,
    num_epochs,
    device="cuda" if torch.cuda.is_available() else "cpu",
):
    # move the
    model.to(device)
    best_val_loss = float("inf")
    for epoch in range(num_epochs):
        # Training Phase
        model.train()
        train_loss = 0.0
```

Successfully trained

```
Epoch 59/75, Train Loss: 10.2340, Val Loss: 10.2677
Epoch 60/75: 100%|██████████| 380/380 [03:27<00:00,  1.83it/s, Train Loss=3.52]
Epoch 60/75, Train Loss: 10.2359, Val Loss: 10.2677
Epoch 61/75: 100%|██████████| 380/380 [03:29<00:00,  1.82it/s, Train Loss=3.54]
Epoch 61/75, Train Loss: 10.2357, Val Loss: 10.2677
Epoch 62/75: 100%|██████████| 380/380 [03:29<00:00,  1.81it/s, Train Loss=14.9]
Epoch 62/75, Train Loss: 10.2346, Val Loss: 10.2677
Epoch 63/75: 100%|██████████| 380/380 [03:28<00:00,  1.82it/s, Train Loss=3.55]
Epoch 63/75, Train Loss: 10.2353, Val Loss: 10.2677
Epoch 64/75: 100%|██████████| 380/380 [03:28<00:00,  1.82it/s, Train Loss=3.54]
Epoch 64/75, Train Loss: 10.2342, Val Loss: 10.2677
Epoch 65/75: 100%|██████████| 380/380 [03:18<00:00,  1.92it/s, Train Loss=3.54]
Epoch 65/75, Train Loss: 10.2340, Val Loss: 10.2677
Epoch 66/75: 100%|██████████| 380/380 [03:17<00:00,  1.92it/s, Train Loss=3.57]
Epoch 66/75, Train Loss: 10.2356, Val Loss: 10.2677
Epoch 67/75: 100%|██████████| 380/380 [03:14<00:00,  1.95it/s, Train Loss=23.6]
Epoch 67/75, Train Loss: 10.2350, Val Loss: 10.2677
Epoch 68/75: 100%|██████████| 380/380 [03:16<00:00,  1.94it/s, Train Loss=3.55]
Epoch 68/75, Train Loss: 10.2353, Val Loss: 10.2677
Epoch 69/75: 100%|██████████| 380/380 [03:11<00:00,  1.98it/s, Train Loss=14.9]
Epoch 69/75, Train Loss: 10.2347, Val Loss: 10.2677
Epoch 70/75: 100%|██████████| 380/380 [03:12<00:00,  1.98it/s, Train Loss=8.28]
Epoch 70/75, Train Loss: 10.2332, Val Loss: 10.2677
Epoch 71/75: 100%|██████████| 380/380 [03:11<00:00,  1.98it/s, Train Loss=8.26]
Epoch 71/75, Train Loss: 10.2355, Val Loss: 10.2677
Epoch 72/75: 100%|██████████| 380/380 [03:13<00:00,  1.96it/s, Train Loss=15]
Epoch 72/75, Train Loss: 10.2344, Val Loss: 10.2677
Epoch 73/75: 100%|██████████| 380/380 [03:13<00:00,  1.97it/s, Train Loss=3.56]
Epoch 73/75, Train Loss: 10.2340, Val Loss: 10.2677
Epoch 74/75: 100%|██████████| 380/380 [03:12<00:00,  1.97it/s, Train Loss=3.55]
Epoch 74/75, Train Loss: 10.2343, Val Loss: 10.2677
Epoch 75/75: 100%|██████████| 380/380 [03:14<00:00,  1.96it/s, Train Loss=8.24]
Epoch 75/75, Train Loss: 10.2350, Val Loss: 10.2677
```

Additional Stuff

- using GPT w/ Pyppeteer to search and then determine R0 and mortality values values

```
112
113
114  def main():
115      try:
116          with open("Virus List.txt", "r") as virusList, open("output.csv", "w") as f:
117              w = csv.writer(f)
118              for line in virusList:
119                  virus = line.strip()  # remove newline characters
120                  print(f"ADDING VIRUS {virus}")
121                  ans = switch_to_csv(getGPTAnswer(virus), virus)
122                  print(ans)
123                  w.writerow(
124                      ans
125                  )  # assuming this function is defined elsewhere
126                  print(f"ADDED VIRUS")
127      except Exception as e:
128          print(f"An error occurred: {e}")
129
130
131  if __name__ == "__main__":
132      main()
```

Additional Stuff

- creating github workflows to organize and standardize our work

```yaml
name: Install Python Dependencies and Check for Conflicts

on:
  push:
    branches:
      - main
  pull_request:
    branches:
      - main

jobs:
  install_and_check:
    runs-on: ubuntu-latest

    strategy:
      matrix:
        python-version: [3.9, 3.12]

    steps:
    - name: Checkout repository
      uses: actions/checkout@v2

    - name: Set up Python 3.12.4
      uses: actions/setup-python@v2
      with:
        python-version: 3.12.4

    - name: Install pip dependencies
      run: |
        python -m pip install --upgrade pip
        pip install -r requirements.txt

    - name: Check for Python dependency conflicts
      run: |
        pip check

    - name: Run tests
```

Model Tests:

- Flu
  - Real Number:
  - Model Predicted Number:
-
HPV
  - Real Number:
  - Model Predicted Number:

TAKEAWAYS:

- discuss loss and accuracy
- We used a different number for I and U in the vocab for our dictionary, we could change that and treat them like analogs to see if it procures different results