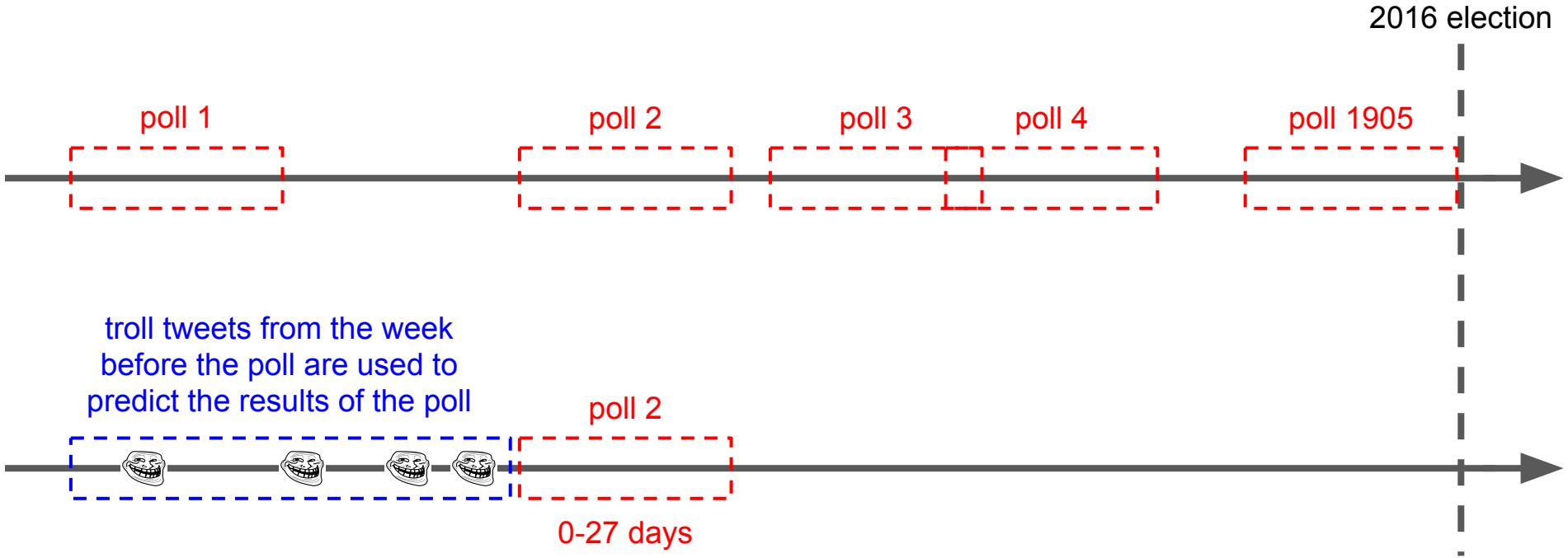
What is the toll of the Russian Twitter troll?

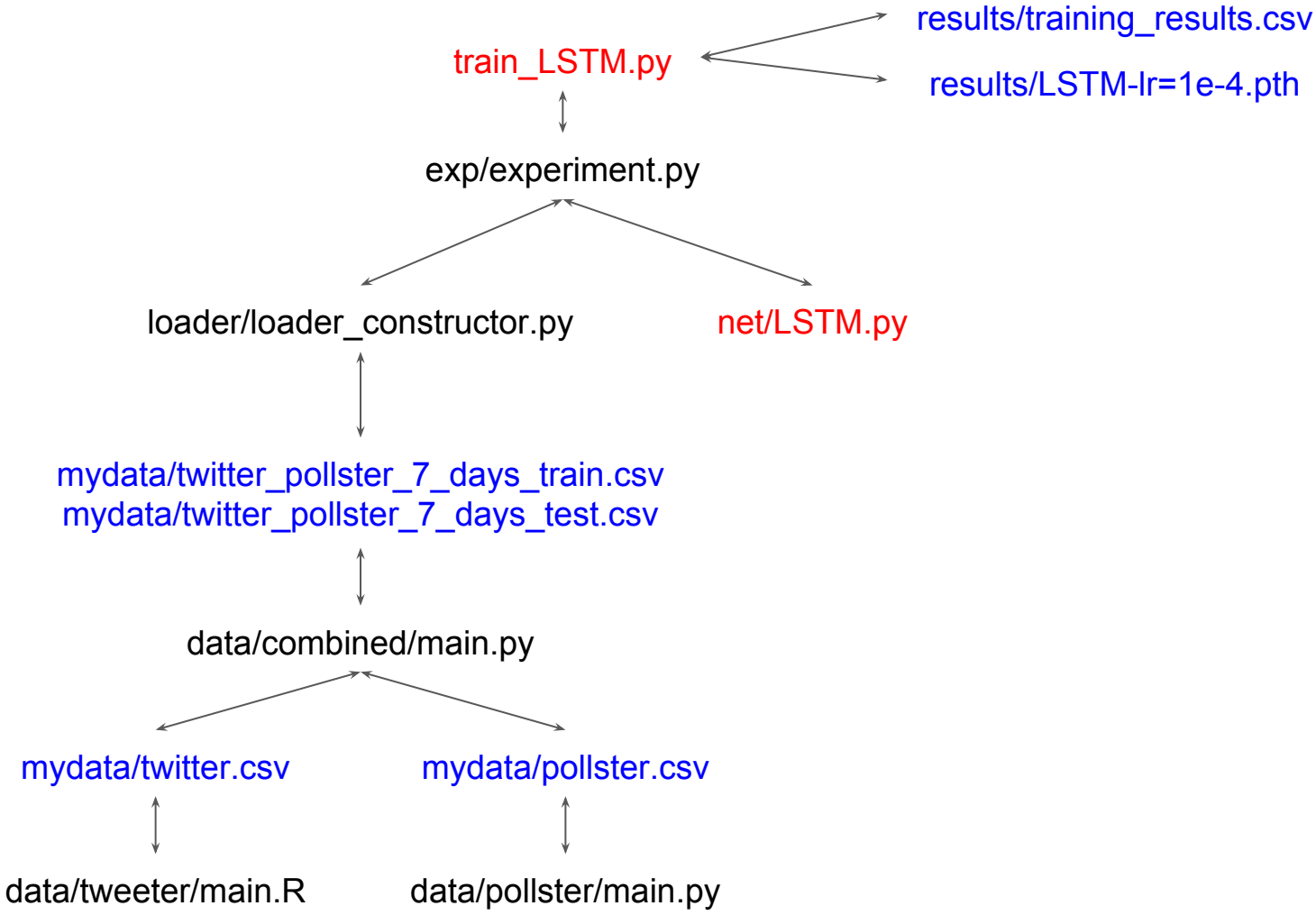
Project goal:

Can we predict poll results from troll tweets written **prior** to the beginning of the poll?

Timeline


2016 election
poll 1
poll 2
poll 3
poll 4
poll 1905
troll tweets from the week before the poll are used to predict the results of the poll
poll 2
0-27 days

Overview


train_LSTM.py
results/training_results.csv
results/LSTM-lr=1e-4.pth
exp/experiment.py
loader/loader_constructor.py
net/LSTM.py
mydata/twitter_pollster_7_days_train.csv
mydata/twitter_pollster_7_days_test.csv
data/combined/main.py
mydata/twitter.csv
mydata/pollster.csv
data/tweeter/main.R
data/pollster/main.py

Dataset 1: Russian Twitter Trolls

Rows: 3 million tweets
Columns: text of tweet

GitHub repository: https://github.com/fivethirtyeight/russian-troll-tweets

File: ProjectTroll/data/tweeter/main.R

Downloads and preprocesses the data by:

- Selecting a subset of tweets
- Filtering non-english

Dataset 2: Pollster

Rows: 1905 polls
Columns: Trump, Clinton

GitHub repository: https://github.com/huffpostdata/python-pollster

File: ProjectTroll/data/pollster/main.py

Downloads the data, no preprocessing needed

Combined dataset

Rows: 702 polls
Columns: concatenation of tweets, Trump, Clinton

File: ProjectTroll/data/combined/main.py

- Assigns to polls tweets from the preceding week
  - Students should check if 1 week is indeed the correct time period
  - A tweet that belongs to multiple polls is assigned randomly to one of them
  - A poll that has no tweets is removed from the dataset
- Splits the polls into 561 training polls and 141 testing polls

Loader

File: ProjectTroll/loader/loader_constructor.py

Uses a PyTorch NLP module called **torchtext** to:

- Tokenize the tweets
- Convert them into lowercase
- Embed the words using Glove

For more detail on torchtext:
http://mlexplained.com/2018/02/08/a-comprehensive-tutorial-to-torchtext/
http://anie.me/On-Torchtext/
https://towardsdatascience.com/use-torchtext-to-load-nlp-datasets-part-i-5da6f1c89d84

Network

Each student implements a different method:

1. Baseline (Lasso, ridge, logistic regression)
2. RNN
3. LSTM: implemented as an example but one student can still choose it
4. LSTM + Attention
5. Self Attention
6. CNN
7. RCNN

Students can also choose other architectures

GitHub: https://github.com/prakashpandey9/Text-Classification-Pytorch

Experiment

File: ProjectTroll/exper/experiment.py

Trains the network

Saves the XYZW array:
results/training_results.csv

Saves the model:
results/LSTM-lr=1e-4.pth

Experiment specification

```
net_list        = [
                  'LSTM',
                  ]

lr_list         = [
                  1e-4,
                  ]

for net_idx in range(1):
    for lr_idx in range(1):

        loader_opts  = {'data_path'                : '/Users/vpapyan/mydata',
                        'days'                     : 7,
                        'Glove_name'               : '6B',
                        'embedding_dim'            : 300,
                        'fix_length'               : None,
                        }

        net_opts     = {'hidden_size'              : 256,
                        'output_size'              : 2,
                        }

        train_opts   = {'crit'                     : 'MSELoss',
                        'net'                      : net_list[net_idx],
                        'optim'                    : 'Adam',
                        'weight_decay'             : 5e-4,
                        'optim_kwargs'             : {},
                        'epochs'                   : 100,
                        'lr'                       : lr_list[lr_idx],
                        'milestones_perc'          : [1/3,2/3],
                        'gamma'                    : 0.1,
                        'train_batch_size'         : 2**7,
                        'test_batch_size'          : 2**9,
                        'device'                   : get_device(),
                        'seed'                     : 0,
                        }

        results_opts = {'training_results_path'    : './results',
                        'train_dump_file'          : 'training_results.json',
                        }
```

Students' Task Until the Hackathon

1. Run the following files:
   File: ProjectTroll/data/twitter/main.R (requires dplyr)
   File: ProjectTroll/data/pollster/main.py (pip install pollster)
   File: ProjectTroll/data/combined/main.py
2. Run ProjectTroll/train_LSTM.py to verify datasets were created correctly (pip install torchtext)
3. Implement your chosen network architecture
   File: ProjectTroll/network/**NETNAME**.py
4. Train network on different hyperparameters
   File: ProjectTroll/train_**NETNAME**.py
5. Once finished, email training_results.csv to all the mentors

Recommended tweaks

- Use full data:
  - Currently we are using 5 files and 2,000 lines per file
  - Instead, use all twitter csvs (13 files) and all lines
- Remove stopping words (“and”, “or”, “are”, ...)
- Remove hashtags, links, rare words…
- Include side information from the twitter data (number of followers, followees, retweets, …)

Recommended tweaks

- Days prior to poll (currently 7 days)
- Changing regression setting to binary classification (renormalize poll results to sum to one and change loss from MSE to cross entropy)
- Parameters of GloVe (dimension, type, etc.)
- Use word2vec instead of GloVe
- Network (width, depth, etc.)
- Optimization:
  - Algorithm (SGD, ADAM, etc.)
  - Learning rate
  - Epochs
  - Epochs in which learning rate drops (gamma)
  - Batch size