Approach 1: MIT Research

1. Data generation using regex and additional logic
2. Two datasets - df_train and df_2
3. Features vector - CountVectorizer (what additional we can do: replace with BioBERT)
4. Prepare Combined trainDataset, validDataset
5. Model :
    a. Test_hyperparamaters : outdict{CombinedNet, FeatureNet, TextNet}
    b. Finetune_bert using BertForSequenceClassification

Reference Github:
https://github.com/Sanger2000/Predicting-Lung-Cancer-Disease-Progression-from-CT-reports

Data Preparation:

Data Description:
Final OUTCOME/Labels for Classification:

- progressive of disease (POD),
- stable disease (SD),
- partial response (PR),
- complete response (CR)

Codeflow:

data_utiles.py
make_dataset(args) --> return trainDataset, validDataset
- create_data [make_features.py]
    - load_reports [preprocess_data.py]
        - make_POD
        - bert_text_cleaner
    - extractFeatures [preprocess_data.py]
        - days_after_start
        - findGroups
            - pruneOverall
                - pruneVols
                - pruneLens
            - extractContext
    - setupFeatureVectors [make_features.py]
        - pad_vectors
        - make_id
        - one_hot_encode
            - prepare_y
    - extractText
    - createTextFeatures [make_features.py]
        - learn_bow
    - tokenizer : Bert tokenization [make_features.py]
        - tokenize_input
    - id_vals [make_features.py]

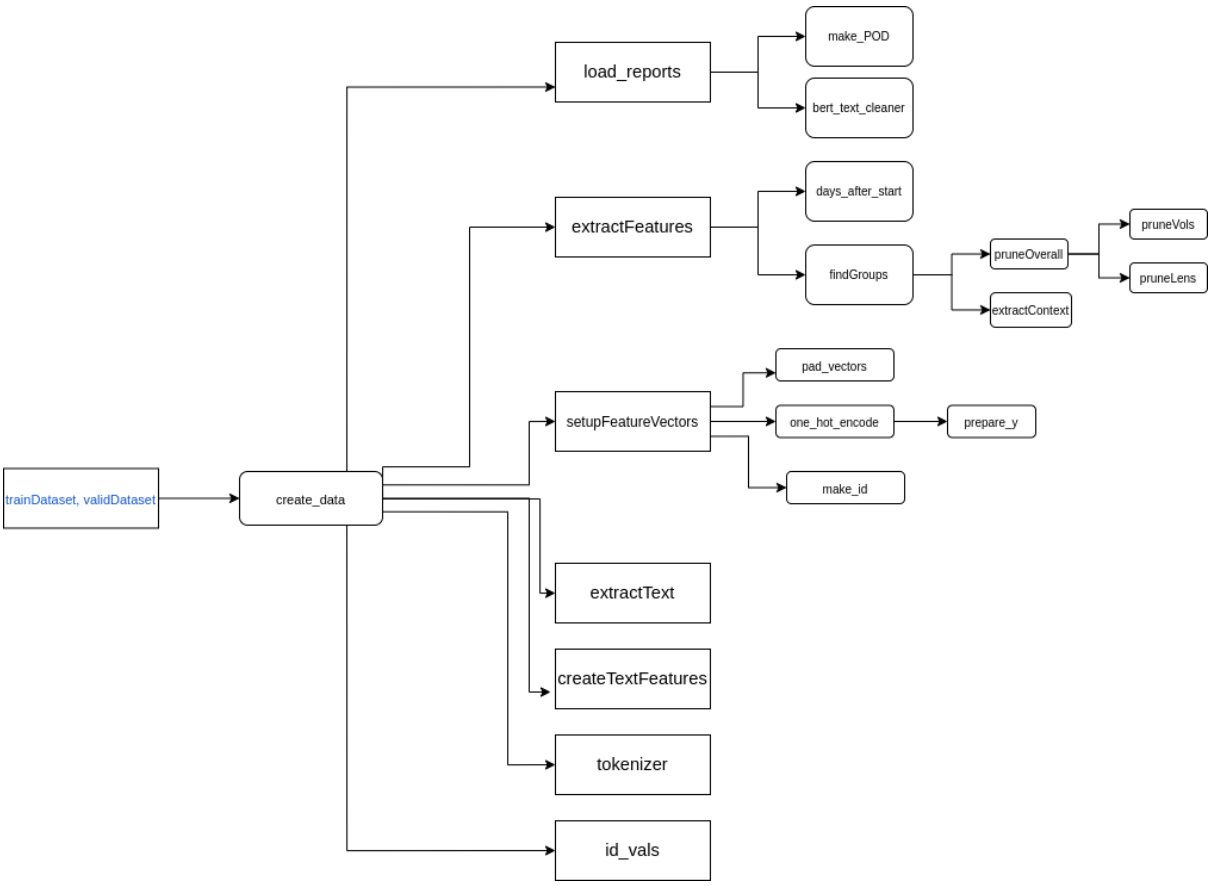

Fig1: Codeflow

	Patient ID	Treatment setting	Objective Response per RECIST v1.1	Treatment start date	Date of radiologic progression-free survival (PFS, calculated from start date)	PFS censor (1 = progressed, 0 = has not progressed)	Scan timepoint (baseline = prior to treatment start, ontx = during treatment or prior to progression if stopped treatment , progression = time of RECIST defined progression)	Scan included on RECIST form? (y/n)	Date of scan	Type of scan	Scan type specified	Scan report text
0	MSK_008	clinical trial	POD	5/17/12	7/12/12	1	baseline	yes	5/2/12	CT	CT CH/ABD/PEL W/ CON	CT CH/ABD/PEL W/ CON\n**ID-NUM\n \n**DATE[MM D...
1	MSK_008	clinical trial	POD	5/17/12	7/12/12	1	progression	yes	7/12/12	CT	FOREIGN CT CHEST - CD	FOREIGN CT CHEST - CD\n**ID-NUM\n \n**DATE[MM ...

Fig2: Training data

Steps:

Preprocessing:

Step1:

1. Consider only RECIST scan observations only
    a. Remaining data: 1775 observations
2. clean text - Notes
    a. - remove extra spaces
    b. - lowercase
    c. - remove special characters
3. Bert_text_cleaner
    a. Extract text between **FINDINGS?:** and **IMPRESSIONS?:**

Output of 1,2 and 3:

- df_train which has 2 columns
    - clean_report_text
    - bert_text : Text between FINDINGS?: and IMPRESSIONS?:

Step2:

4. ExtractFeatures
    a. findGroups
    {"before_text":[], "after_text":[], "organs":[], "lens":[], "date_dist":[], "timepoint":[], "Patient ID":[], "labels": []}
    '''Input text from Scan Text Report
    Output: A dictionary for each organ of the text involved in it '''
        i. pruneOverall
        Input: dictionary of organ descriptions
        Output: returns dictionary of organ descriptions where each description has volume or length measurements replaced also returns dictionary of organ tumor minor axis measurements
        ii. pruneVols
        Input text of a given organ description
        Output: extracts the smaller axis lengths from a given volume measurement and replaces it with generic aabbSIZEbbaa word
        iii. pruneLens
        Input text of a given organ description
        Output: extracts the lengths and replaces it with generic aabbsizebbaa word

Organs:

- Extract organs : Ex. LUNG, PLEURA/PERICARDIUM
- Label Encoding (1:N)

Lens:

- (\d\.?\d?) ([CcMm][Mm])
- mult = {"cm": 10, "mm": 1}
- Input text of a given organ description
- Output: extracts the lengths and replaces it with generic aabbsizebbaa word
- Example : extracted 4cm --> 4*10 = 40cm

```
LUNGS:                                  A poorly defined confluent right
hilar mass is grossly stable and measures approximately 5.1 x 4 cm,
previously 5.2 x 3.8 cm. Consolidative findings are seen extending from
the right hilum to the anterior right upper lobe, right lower lobe, and
the right middle lobe. Numerous tiny pulmonary nodules are seen throughout
the lungs bilaterally. Slightly increased septal markings are noted in the
upper lung zones.

PLEURA/PERICARDIUM:          Stable small to moderate right pleural
effusion. A minimal left pleural effusion is also again noted. There is a
small pericardial effusion which is stable.

THORACIC NODES:               A slightly enlarged left supraclavicular
lymph node measures 1.5 x 0.8 cm, previously 1.2 by 0. there on multiple
small to slightly enlarged mediastinal lymph nodes similar to prior
examination. A subcarinal lymph node measures 1.4 x 1 cm. In addition to
the poorly defined confluent right hilar mass there is mild left hilar
adenopathy, difficult to compare to prior noncontrast examination.

HEPATOBILIARY:                   A small heterogeneous nodule is noted
laterally at segment 5/6, and this questionably was present on **DATE[MM DD
2012], but comparison is not possible due to the absence of intravenous
contrast on the prior examination. Some areas of perfusion anomaly are
noted at hepatic segment 4. Cholelithiasis is present. No dilatation of
the biliary tree is present.

SPLEEN:                                 unremarkable

PANCREAS:                             unremarkable
```

Fig3: ex of between text

	before_text	after_text	organs	lens	date_dist	timepoint	Patient ID	labels	is_baseline
0	stable and measures approximately	, previously aabbSIZEbbaa .	18	45.5	-0.789903	baseline	MSK_008	POD	True
1	, previously	.	18	45.0	-0.789903	baseline	MSK_008	POD	True
2	supraclavicular lymph node measures	, previously 1.	41	11.5	-0.789903	baseline	MSK_008	POD	True
3	subcarinal lymph node measures	.	41	12.0	-0.789903	baseline	MSK_008	POD	True
4	6 image 200 measuring	.	30	10.5	-0.789903	baseline	MSK_008	POD	True
5	mass has increased previously	now aabbSIZEbbaa .	18	45.5	-0.504963	progression	MSK_008	POD	False
6	now	.	18	56.0	-0.504963	progression	MSK_008	POD	False

Fig4: df_2 - data after performing ExtractFeatures

Step3:

4. setupFeatureVectors

Input:

- df (output of ExtractFeatures)
- Desired_features = ("lens", "organs", "date_dist")
- Max_before = 600
- Max_after = 300
    a. learn_bow: on before_text and after_text
        i. Remove stopwords = ['mm', 'dd', '2017', '2016', '2015', '2014', '2013', '2012', 'date', 'md']
        ii. CountVectorizer.transform(reports) (Word Embeddings)
    b. Train_features: pad_vectors(zero padding)

Output: baseX=train_features[False]), progX=train_features[True], labs=one_hot_encode, id_list= patient id

5. extractText

Input: df_train, id_list

- is_baseline= {True , False}
- Baseline_text : groupped_text is_baseline== True

- Progress_text : groupped_text is_baseline==False

Output:
baseline_text, progress_text, baseline_bert, progress_bert

6. tokenization

Input: baseline_text, context_text, max_len=509, tokenizer=bert-base-uncased

c. Tokenizer - used bert tokenizer
d. Id_vals

[0, 0] + [0]*baseline tokens + [1] + [1]*context tokens + [0]*max_len-(len(context) + len(baseline))

Final token = "[CLS]" + baseline tokens + "[SEP]" + context tokens + "[SEP]" + "[MASK]"

Output:
Id_vals = tokenizer.convert_tokens_to_ids(final_tokens)
Id_vals[:,0,:]
id_vals[:, 1,:]

Step 4:

7. Create_data:

a. load_reports
b. extractFeatures
c. setupFeatureVectors
d. extractText
e. createTextFeatures
   i. Ip- reports(baseline_text{clean_text where is_baseline=False), progress_text)
   ii. Op- overallTextFeatures (df_text) : sequence horizontally
f. tokenizer
g. id_vals

Input:
max_base=400, max_prog=800, max_before=600, max_after=300,
desired_features=("lens", "organs", "date_dist")
Output:
All the following output is groupby patient_id

baseX=train_features[False]) -> setupFeatureVectors -> df_2 -> is_baseline = False

- Combine before_text and after_text based on is_baseline = False

progX=train_features[True] -> setupFeatureVectors -> df_2 -> is_baseline = True

- Combine before_text and after_text based on is_baseline = True

labs=one_hot_encode -> setupFeatureVectors -> labs = one_hot_encode(train_labels)

Df_text -> createTextFeatures -> overallTextFeatures

- Used df_train -> clean_report_text

- Add/hstack - learn_bow(baseline_text) + learn_bow(progress_text)

Id_vals[:,0,:] - token_ids -> id_vals
- Bert_text - is_baseline=True

id_vals[:, 1,:] - segment_ids -> id_vals
- Bert_text - is_baseline=True

Step 5:

8. Make_datasets:
    a. Create_data
    b. Split data into train(0.8) and validation(0.2) data

Output: trainDataset, validDataset

Model

Model_utils.py:

1. **TextNet**

X -> Linear(X) -> relu -> dropout -> softmax

2. **FeatureNet**

x, y ->
Shared_layer = Linear()
First_base = relu(shared_layer(x))
First_progress = relu(shared_layer(y))
Overall_out =Linear(torch.cat(First_base, First_progress)
softmax(Overall_out)

3. **CombinedNet**

x, y, z ->
Shared_layer = Linear()
First_base = relu(shared_layer(x))
First_progress = relu(shared_layer(y))
Overall_out =Linear(torch.cat(First_base, First_progress, z)
softmax(Overall_out)

Main.py

1. Pass args
2. Test_hyperparamaters

    a. train_data, valid_data
    b. Model_concat.pt : model1= CombinedNet
    c. Model_features.pt : model2 = FeatureNet
    d. Model_text.pt : model3 = TextNet
3. Finetune_bert
    a. train_data, valid_data
    b. BertForSequenceClassification
    c. Bert_model.pt : final fine-tuned model
4. Loss used: mse_loss