PROGRAMMER MANUAL

This describes the "how" part (changes, commenting and removing comments for different approach, etc.) of implementing the various python scripts in the "Code" directory. Please read this manual along with the user manual; both have the same order of instruction and complement each other.

'Tweet-Snippet Interchange': For interchanging between tweets and snippets, just interchange between the words 'tweets' and 'quotes' in every file name for every script. Also change the initial path(if any) between 'tweets.xlsx' and 'quotesA+V.xlsx' (or snippets.xlsx, please change file names if required!). HOLDS FOR ALL SCRIPTS.

To implement rule based extractor –

1. **rules_implement.py** -> DO NOT RUN.
   - This implements all the 9 rules in separate functions (R1-R9).
   - Modify existing rule implementation or add new rule here.
   - This is imported in the next script.
2. **rule_weighting.py** ->
   - By default, this script will run for conditional results for all rules.
   - For normal performance results :
     i) Change 'conditional' word in file names to 'normal'
     ii) Divide by 'denom' instead of 'n' (use acc1,acc2,acc3 instead of acc4,acc5,acc6)

3. **rule_based_app.py** -> only Tweet Snippet Interchange required.
4. **rule_based_app_evaluation.py** ->
   - Above script has given us the candidate targets for the 3 metrics for each dataset, and now these are compared with the actual target to evaluate rule based extractor.
   - Results for majority and weighted majority approach are printed and not stored.
   - Script can be run directly for tweets (interchange required for snippets).

To implement statistical extractor –

(For **sequence labeling** - > suitable directory name changes + do not run step 2)

1. **generate.py** - >
   - 17 feature combinations (t1-t17) imply 17 train and test files need to be produced for each of the 4 folds.
   - Feature combination can be set by the Boolean array 'bf'. The 7 flags in this array represent a feature each, described in the generate_train_file() function.
   - Set corresponding train and test file name (t1-t17) according to above setting.
   - run_flag sets the fold (1-4)

   1 supporting function – **tag(),** takes in a text and produces corresponding pos tag sequence.

2 main functions: **generate_train_file()** and **generate_test_file()** to produce the train and test files respectively.

2. **convert_for_svm_labels.py** -> Changes required for the two file names – the apt run folder (run1-run4) and the apt test file no. (1-17)
3. **generate_sents.py** - > only Tweet Snippet Interchange required.
4. **Run_SVM.txt** -> implement the command on command line, and follow given instructions. After this, we have all the predictions and files ready for evaluation of this statistical extractor.
5. **classification_acc.py** - >
   - not for running – a function used by the next script.
   - classification_tweets <-> classification_quotes for the 4 files.
   - Apart from this, by default, script is set to produce the harmonic_res files (for dice score), (refer README1 file in results).
   - To get the 'res' files (for partial and exact match) ->
     i) change file name of the last file.
     ii) Let the variables [part ; full ; partio ; percs ; partacc ; fullacc] run their course instead of [hm ; hmean]
6. **produce_classi_results.py** ->
   - Basically a nested loop to implement above script for all possibilities (for all 4 folds and for all 17 feature combinations)
   - Import and use classification_acc.py (instead of integ_34.py)
7. **final_results.py** -> (set for hybrid approach by default).
   To convert for svm (statistical), please make the following changes for –
   a) partial and exact match results – (result files named 'results_t') check the apt files f1,f2. Required variables – pa, fa, percs,x,y,z.
   b) Only dice score results – (result files named 'harmonic_results_t') check fapt file names (f1,f2) . Required variables – hm,v.

<u>To implement hybrid framework –</u>

1. Get the required result files of the **rule base extractor -> steps 1-2**
2. Get the required result files of the **statistical extractor -> steps 1-4**
3. **integ_34.py** ->
   - not for running – a function used by the next script
   - classification_tweets <-> classification_quotes for the 6 files.
   - In one run this produces all 3 files (for the 3 metrics) for either tweets or quotes and for either hybrid AND, or hybrid OR
   - Hybrid AND set by default
   - To change to OR ->
     i) Replace 'AND' by 'OR' in every file name.
     ii) Change 'intersection' to 'union' for sets (lines 108-110)
   - File fg to store correctly detected outside cases (list indices).
   - All results are stored in files, but outside case results are printed.

4. **produce_classi_results.py** ->
    - Basically a nested loop to implement above script for all possibilities (for all 4 folds and for all 17 feature combinations)
    - Import and use integ_34.py (instead of classification_acc.py)
5. **final_results.py** - >
    - set for hybrid AND by default.
    - For hybrid OR, just change the AND in all the file names to OR.