NLG Workshop using TGen

- Introduction

  This workshop is adopted from the github project UFAL-DSG/tgen

  https://github.com/UFAL-DSG/tgen/blob/master/USAGE.md#experiments

  The experiment is conducted on Alex NLG dataset

- Installation (Admin could be required if on Windows C drive)
  a. For windows Please install ***git*** from https://git-scm.com/download/win
  b. Create a new Env (e.g., NLGENV) in Anaconda
  c. Active the ENV from command line :
     - conda activate NLGENV
  d. Download and unzip ***tgen-master*** from Luminus and access the folder *tgen-master*
  e. Install requirements

     #TGen Day3 AM
     pip install regex
     pip install unicodecsv
     pip install enum34
     pip install rpyc
     pip install numpy==1.18.1
     pip install tensorflow==1.15.2
     pip install pudb
     pip install recordclass
     pip install https://github.com/kpu/kenlm/archive/master.zip
     pip install git+https://github.com/ufal/pytreex

- Data Preparation
  1. Verify ***dataset.json*** in the folder of \tgen-master\alex-context\input\

  2. Run the script of \tgen-master\alex-context\input\convert.py to convert the dataset into training and testing in the format desired by **TGen**
     - python convert.py dataset.json train.data,test.data -s 4:1

     - Verify Dialogue Acts (DAs) files and Text files are generated:

       X_train: alex-context\input\train.data-das.txt
       Y_train: alex-context\input\train.data-text.txt
       X_test: alex-context\input\test.data-das.txt
       Y_test: alex-context\input\test.data-text.txt

     - **X_train** and **X_test**: The main input format into TGen are lists of triples of the shape (DA type, slot/attribute, value),
       e.g.: inform(food=Chinese)&inform(price=expensive).

- DAs are delexicalized in a typical case.

```
inform_no_match(from_stop=X-from_stop)
inform_no_match(from_stop=X-from_stop)
inform_no_match(from_stop=X-from_stop)
iconfirm(to_stop=X-to_stop)
iconfirm(to_stop=X-to_stop)
iconfirm(to_stop=X-to_stop)
request(from_stop)
request(from_stop)
request(from_stop)
request(from_stop)
request(from_stop)
request(from_stop)
inform_no_match(to_stop=X-to_stop)&inform_no_match(departure_time=X-departure_time)
inform_no_match(to_stop=X-to_stop)&inform_no_match(departure_time=X-departure_time)
inform_no_match(to_stop=X-to_stop)&inform_no_match(departure_time=X-departure_time)
```

- **Y_train** and **Y_test**: Outputs plan text for direct string generation. Use one output sentence per line (no comments/empty lines allowed). For best results, delexicalize sparse values, such as restaurant/landmark names, time values etc. and fill them in in a postprocessing step

```
Sorry , your trip from X is not found .
I did not find a route from X .
Not found from X .
You want to go to X .
You want to go to X .
OK , you want to go to X .
From what station ?
From where ?
Where are you traveling from ?
Yes , but from where to go ?
OK sir but from where do you want to go ?
Please confirm where you are departing from ?
Not found at X to X .
Apologies , a connection to X at X is not found .
Sorry sir but I could not find any transport to X at X .
```

3. Check the configuration script : alex-context\config\seq2seq.py
   -

- Train the Model
  4. python run_tgen.py seq2seq_train alex-context\config\seq2seq.py alex-context\input\train.data-das.txt alex-context\input\train.data-text.txt alex-context\model.pickle.gz

- Test the Model
  5. python run_tgen.py seq2seq_gen -w alex-context\out-text.txt alex-context\model.pickle.gz alex-context\input\test.data-das.txt