milestone3

June 16, 2021

1 Milestone 3

Stephanie Eordanidis. Ravjot Sachdev, Jackson Taber
Syracuse University : College of Engineering & Computer Science
223 Link Hall, Syracuse, NY 13244
sleordan@syr.edu, rssachde@syr.edu, jrtaber@syr.edu
CIS 700 Machine Learning and Security
06/16/2021

1.1 Theme:

"Adversarial Text Generation: Adversarial Machine Learning Applications in Text Analysis"

1.2 Purpose:

The purpose of this lab is to add three new GAN metrics to the project space and successfully run them on the chosen GAN models from previous milestones.

1.3 Project:

Texygen is the name of the project selected. This project is a benchmarking tool that aids in text generation model research and testing. This tool allows for ease of various model testing to compare accuracy and synthetic data generation of models using same training baseline.

1.4 (Hard/Soft)ware:

Google Colaboratory https://colab.research.google.com/ **GPU** Python 3 Google Compute Engine backend **Github** https://github.com/eordanis/CIS-700_clone/

1.5 Resources:

Original Source: https://github.com/geek-ai/Texygen
Modified Sources Acquired: 2SU Course Files Section -> Texygenmaster_Python_3.6.zip

1.6 Data:

The data for the selected project is setup as follows: Generated data training: 5000 word and 20 sentence count Oracle data generation: 10,000 sentence generation Real data training:

- image_coco : 20,000 sentences chosen from the image

- COCO captions data. 10,000 of which are used as training set while other 10,000 used as test set
- emlp_news_min : 20,000 sentences
- A minified version of emlp_news : reduced from 278,586 lines of text in training data to 10,000 and also 10,000 for test. Trying to run real training on 1/4 of a million lines proved too taxing even on colab pro. minified version should yeild decent results comparable to image_coco, the project default.
- eapoe : 266 sentences
- eapoe : 266 sentences Compiled from various Edgar Allan Po Poems found on referenced poem sight [4].

1.7 Modifications:

To begin, the modified source code acquired from the 2SU application was further modified to combine the original intention of the origin source authors as well as professor modification. If no arguments are passed, all models/data are run. > **Note**: The order or model generation is done by first iterating over the GAN model type, then data type. If arguments are passed, those arguments will be validated and run accordingly to run a more targeted model test.

Modifications were applied to eliminate much library warnings and informational messages as to keep output as clean as possible. File path naming was updated to be compliant with Google Colaboratory environment. All epoch time elapse console printing has been commented out for cleaner output reading.

Modifications under previous course works: * Addition of three new GAN models * CGAN * InfoGAN * DCGAN

- Fixed LeakGAN project to work and run properly by setting standardized default flag attributes in main.py
- Update models to save off test files for each training type run (ie oracle, cfg, real)
- Updated models to use unified naming schema for test files, with model as the model the file was generated for and training as the training the data was generated from **Example:**
  - experiment-log-model-training.csv
  - oracle-model-training.txt
  - generator-model-training.txt
  - test_file-model-training.txt
- Updated models to set file name in main.py on the GAN model directly instead of in each model file themselves
- Added additional field called 'log_file' to be the name for the experiment-log file data
- Update to add new dir for midterm content vs milestone project
- Added new util called visual.py to handle data representation
- Allow visual.py to take directory param. if none exist, default to /content/CIS-700_clone/results

- Updated visual.py to grab files from directory and generate visual data representations from relevant files in a more automated and less hard coded way
- Allow main.py to take a results output directory via arg -o, if arg not present sets to results/ • Allow main.py to take value for epoch via -p arg to use in model training
- For argument provided training, added more detailed messaging as well as metric grid showing metrics that will be used during model generation
- Cleaned main.py and model files to limit print to console, additionally console print is uniform and consistent across.
- Added time elapsed pring statement for model run.

Modifications under this work: * Addition of three new metrics: * Time Elapse Interval (TEI) TEI measures time elapse for the model per epoch run. Used to measure average time efficiency to run model for each training during epochs. Helps gauge run time during various model training components (ie pre training and adversarial training) * Accuracy (ACC) ACC is implemented as a discriminator metric that measures the accuracy of the discriminator's predictions on the labelled input data. A higher accuracy indicates that the discriminator's predictions align with the input data. * Perplexity (PPL) PPL is implemented as a generator metric that measures a model's confidence in its next prediction, with lower perplexity scores indicating a stronger confidence of the model's prediction.

1.8 Setup:

Due to the heft of processor/gpu usage, it was deemed necessary to run the project in the Google Colaboratory. Original attempt to run was done via Pycharm IDE Professional Edition with Anaconda derived environments, however this proved too great of a strain on the accessible workstation.

Additionally, it is important to note if timeout is experienced, it is possible to run a ClickConnect script via inspector tools to prevent timeout while running long codes. The following code worked in chrome when inserted in the developer console at time of test:

```
let myClick = function ClickConnect({ console.log('Working - Preventing Timeout'); document.querySelector('colab-connect-button').shadowRoot.getElementById('connect').click(); } setInterval(myClick,60000);
```

1.8.1 Step 1

A new Google Colaboratory workspace was setup, titled "Milestone3". This workspace was run using the hosted runtime environment. This document is the current document being read.

In order to run against provided code base, it was necessary to sync the colab workspace the github repository files as follows

```
!git clone https://$GITHUB_AUTH@github.com/eordanis/CIS-700_clone/
```

Running this command from the first cell in the workbook syncs the drive to the github repo location of project location, as well as change to the necessary directory

```
[ ]: import shutil
     # to prevent nesting problems, remove directory and its contents if exists
     #if called to remove known existing, must restart runtime before cloning again
     dir_path = '/content/CIS-700_clone'
     try:
         shutil.rmtree(dir_path)
     except OSError as e:
         print("Error: %s : %s" % (dir_path, e.strerror))
```

```
[ ]: !git clone https://$GITHUB_AUTH@github.com/eordanis/CIS-700_clone/
     %cd CIS-700_clone
```

1.8.2 Step 2

Now it was necessary to import and download any missing libraries the hosted colaborartoy runtime did not have readily available via the following commands:

!pip install -r "requirements.txt" import nltk nltk.download('punkt')

Running this command from the next cell in the workbook installed the necessary libraries and at specified versions for the project.

```
[ ]: !pip install -r "requirements.txt"
     import nltk
     nltk.download('punkt')

     !wget http://kheafield.com/code/kenlm.tar.gz
     !tar -zxvf kenlm.tar.gz
     %cd kenlm
     !mkdir -p build
     %cd build
     !cmake ..
     !make -j 4
     %pip install https://github.com/kpu/kenlm/archive/master.zip
     %cd ..
     %cd ..
```

1.8.3 Step 3

Now it is time to run the application. Below are two examples of commands to run the application.

!python3 "main.py"

This first command was run without parameters. In the case of this command, all trainings (SeqGAN, Gsgan, TextganMmd, Leakgan, Rankgan, Maligan, Mle) were run on all available defaulted training data (oracle LSTM, real data, CFG). Running this command can take around 2+ hours to complete.

!python3 "main.py" -g seqgan -t real

This second command was run with parameters. In the case of this command, main was run with SeqGAN training on image_coco. Running targeted trainings take less time to run, on average completing

in 5-15 minutes depending on selected parameters. With the above selection, runtime was run above 10 minutes.

!python3 "main.py" -g seqgan -t real -d data/eapoe.txt

This third command was run with parameters. In the case of this command, main was run with SeqGAN training on eapoe.

!python3 "main.py" -g seqgan -t real -d data/eapoe.txt -o results/test/

This third command was run with parameters. In the case of this command, main was run with SeqGAN training on eapoe and results will output to results/test directory.

!python3 "main.py" -g seqgan -t real -d data/eapoe.txt -o results/test/ -p 45

This fourth command was run with parameters. In the case of this command, main was run with SeqGAN training on eapoe and results will output to results/test directory and run on 45 epoch for both pre and adversarial training .

Running targeted trainings take less time to run, on average completing in 5-15 minutes depending on selected parameters. With the above selection, runtime was run above 10 minutes.

> **NOTE:** For above estimates, based around 5 epochs. Additionally, CFG training appears to have stopped working suddenly, unsure why broken. Therefore running without that option for the time being. Additionally, the LeakGan model failed entirely to run now due to flag errors, so this model was discarded from testing.

1.9 Overview

1.9.1 Process

When running the various models, there are similar steps for each. 1. Beginning Training – begin model training(s) 2. Set training - sets the desired model training method 3. Start model pretrain generator – uses the training data to pre-train the generator model 4. Start model pre-train discriminator - – uses the training data to pre-train the discriminator model 5. Model adversarial training – runs the model to generate results based on the test data and metrics applied 6. Finish Training – end of model training(s)

During training, each model training runs through several passes or epochs. For simplicity, base epoch is set to 5, with model training running thrice for 15 total epochs there abouts for each model trained on a particular data set.

1.9.2 Baseline Models

For this report, the TextGAN and SeqGAN models were run on oracle and real training types in the previous project milestone and will be used as the baseline for new model comparisons. The real training types essentially runs the data against the image_coco.txt caption data. The TextGAN and SeqGAN was developed by the source project team to improve on existing GAN networks.

With regards to TextGan, the goal of this model is to generate high quality realistic synthetic data while overcoming the convergence dilemma by using a generator that runs as a long short-term memory network and its discriminator a convolutional network. By matching highdimension laten feature distributions of the testing and training data, this model over longer epochs has shown demonstrate a higher performance in quantitative evaluation, showing the TextGAN model can produce sentences that appear to have been written by a human, and not AI generated.

For the SeqGAN model, this also proved successful in generating realistic looking sentences via this generator process. A second model was selected for comparison purposes. SeqGAN’s generator is based off

the reinforcement learning stochastic policy, allowing SeqGAN to performing gradient policy update in order to circumvent differentiation issues in the generation. Its discriminator is run on complete sentences, and its results used as the reinforcement learning reward signal. According to source authors, this model boasted higher performance over others run.

1.9.3 New Models From Previous Milestone 1.9.4 CGAN - Conditional Generative Adversarial Network

Conditional adversarial network, or CGAN for short, is a basic modification of GAN that simply adds an additional layer that conditions both the descriminator and generator model layers. For this labs purposes, an existing GAN model was used as the base, and in both the Generator and Discriminator an embedded layer was incorporated as the first layer, used as the conditioning layer. For the sake of specificity, this GAN could actually be considered an CGSGAN, however for simplicity we will refer to is as simply CGAN.

1.9.5 INFOGAN

InfoGan is an adjusted simple version of GAN that seeks to maximize the mutual information of a fixed subset of GAN noise varaibles. It is able to achieve this by having numerous convolution layers added to a regular implemtation of a GAN that are connected at the end, this makes the additional cost of computation low. For the use of this lab an existing GAN was used as the base and the convolution layers were added and connected at the end.

1.9.6 DCGAN

Deep convolutional generative adversarial networks (DCGANs) are a class of convolutional networks (CNNs) aimed to incorporate unsupervised learning. Some key components of the DCGAN architecture are the use of the Tanh activation function for the generator's output layer, LeakyReLU activation in the discriminator, and the removal of any fully connected hidden layers. With these features, the original study conducted was able to create a robust DCGAN, achieving an 82.8% accuracy on the on the CIFAR-10 image dataset.

1.9.7 Metrics

Abbreviations: * BLEU - BiLingual Evaluation Understudy * GAN - Generative Adversarial Network * NLL - Negative Log-Likelihood * RL - Reinforcement Learning

Definitions * EmbSim – influenced by BLEU, used instead to compare the word embeddings vs BLEU's comparison of word similarity between two sentences or documents. * NLL-oracle : applied to synthetic data to determine fitting via oracle language model standards. * NLL-test : dual to NLL-Oracle, used to determine a model's capability to fit to real test data

These measurement standards and more are discussed in the project directory's "/docs/evaluation.md" location.

1.9.8 New Metrics

Time Elapse Interval (TEI) TEI measures time elapse for the model per epoch run. Used to measure average time efficiency to run model for each training during epochs. Helps gauge run time during various model training components (ie pre training and adversarial training)

Accuracy (ACC) ACC is implemented as a discriminator metric that measures the accuracy of the discriminator's predictions on the labelled input data. A higher accuracy indicates that the discriminator's predictions align with the input data.

Perplexity (PPL) PPL is implemented as a generator metric that measures a model's confidence in its next prediction, with lower perplexity scores indicating a stronger confidence of the model's prediction.

1.10 Test

For this project, we will use the following arguments

#directory to use for model results output directory = '/content/CIS-700_clone/results/'

#data set location to use data = '/content/CIS-700_clone/data/eapoe.txt'

#epoch values to run for pre and adversarial training epoch = '5'

Epochs were increased left to run at 5 for the sake of time. However, it is noted that as according to original project sourcing, >= 45 epochs for the models display the best NLL loss results on epochs > 40, prior to that point results would be poorer. NLL loss values are indicated to be better the lower they are, so if these values trend downward, the models are improving. For EmbeddedSimilarity, higher values are desired for better results.

[4]:
```
#data set location to use


# epoch values to run for pre and adversarial training
epoch = '5'
```
data = '/content/CIS-700_clone/data/eapoe.txt'

1.10.1 CGAN

The following commands are to run CGAN model on both oracle and real trainings. **NOTE:**
The real data essentially trained the model on the eapoe.txt data.

!python3 "main.py" -g cgan -t oracle -d $data -p $epoch
!python3 "main.py" -g cgan -t real -d $data -p $epoch

Oracle Training

[5]:
```
!python3 "main.py" -g cgan -t oracle -d $data -p $epoch
```
Unspecified Output Director Location: Defaulting to --> results/

****

Model Set: Cgan
Training Set: Oracle
Data Set: /content/CIS-700_clone/data/eapoe.txt
Epochs Set: 5
Results Output Directory Set: results/

********** Recording

Time Elapse...

******** Beginning Training ******** train_data = results/generator_cgan_oracle.txt test_data = results/oracle_cgan_oracle.txt
Metrics Applied: nll-oracle, nll-test, EmbeddingSimilarity, tei, acc, ppl Pre-training Generator...
epoch:1 nll-oracle:10.534829 nll-test:7.989137
EmbeddingSimilarity:-0.13330428984006423 tei:1.9804988225301108 acc:0 ppl:3273.7598
Pre-training Discriminator... Adversarial Training...
epoch:6 nll-oracle:14.63331 nll-test:7.8258853
EmbeddingSimilarity:-0.1296256958635492 tei:1.6517211516698203 acc:0.59375 ppl:3273.7598
epoch:10 nll-oracle:9.977864 nll-test:8.206214
EmbeddingSimilarity:-0.13391953215792934 tei:3.305062166849772 acc:0.0625 ppl:3273.7598

******** Completed Training ********

Time Elapsed: 07mins:36secs

For below model training and test, results are discussed later. For more details, see Section **??**.

Real Training

[6]:
```
!python3 "main.py" -g cgan -t real -d $data -p $epoch
```

Unspecified Output Director Location: Defaulting to --> results/

****

Model Set: Cgan
Training Set: Real
Data Set: /content/CIS-700_clone/data/eapoe.txt
Epochs Set: 5
Results Output Directory Set: results/
********** Recording

Time Elapse...

******** Beginning Training ******** train_data = results/generator_cgan_real.txt test_data = results/oracle_cgan_real.txt
Metrics Applied: nll-test, EmbeddingSimilarity, tei, acc, ppl Pre-training Generator...
epoch:1 EmbeddingSimilarity:-0.36817533747723885 nlltest:6.361944 tei:0.9378317952156067 acc:0 ppl:1020.2216 Pre-training Discriminator...
Adversarial Training...
epoch:6 EmbeddingSimilarity:-0.3295426012180885 nll-test:6.037096 tei:1.04540722767512

acc:0.9375 ppl:1020.2216 epoch:10 EmbeddingSimilarity:-0.14000588788342286 nlltest:6.9487686 tei:2.530750521024068 acc:0.03125 ppl:1020.2216

******** Completed Training ********

Time Elapsed: 04mins:58secs

For below model training and test, results are discussed later. For more details, see Section **??**.

1.10.2 DCGAN

The following commands are to run DCGAN model on both oracle and real trainings.

NOTE: The real data essentially trained the model on the eapoe.txt data.

```
!python3 "main.py" -g dcgan -t oracle -d $data -p $epoch
!python3 "main.py" -g dcgan -t real -d $data -p $epoch
```

Oracle Training

[7]:
```
!python3 "main.py" -g dcgan -t oracle -d $data -p $epoch
```

Unspecified Output Director Location: Defaulting to --> results/

****

Model Set: Dcgan
Training Set: Oracle
Data Set: /content/CIS-700_clone/data/eapoe.txt
Epochs Set: 5
Results Output Directory Set: results/

********** Recording

Time Elapse...
******** Beginning Training ******** train_data = results/generator_dcgan_oracle.txt test_data = results/oracle_dcgan_oracle.txt
Metrics Applied: nll-oracle, nll-test, EmbeddingSimilarity, tei, acc, ppl Pre-training Generator... epoch:1 nll-oracle:10.720178 nll-test:7.441417
EmbeddingSimilarity:-0.21503435510202246 tei:1.9748214801152548 acc:0 ppl:6726.8167
Pre-training Discriminator... Adversarial Training...
epoch:6 nll-oracle:10.562969 nll-test:7.2489514
EmbeddingSimilarity:-0.21475022843196706 tei:1.9920844078063964 acc:0.4375 ppl:6726.8167
epoch:10 nll-oracle:10.552996 nll-test:7.2919326
EmbeddingSimilarity:-0.20764352832172314 tei:3.8359674493471783 acc:0.46875 ppl:6726.8167

******** Completed Training ********

Time Elapsed: 08mins:28secs

For below model training and test, results are discussed later. For more details, see Section **??**.

Real Training

[8]:
```
!python3 "main.py" -g dcgan -t real -d $data -p $epoch
```

Unspecified Output Director Location: Defaulting to --> results/

****
Model Set: Dcgan
Training Set: Real
Data Set: /content/CIS-700_clone/data/eapoe.txt
Epochs Set: 5
Results Output Directory Set: results/

********** Recording

Time Elapse...

******** Beginning Training ******** train_data = results/generator_dcgan_real.txt test_data = results/oracle_dcgan_real.txt
Metrics Applied: nll-test, EmbeddingSimilarity, tei, acc, ppl Pre-training Generator...
epoch:1 EmbeddingSimilarity:-1.1394602891302366 nll-test:5.6753507 tei:0.9595076004664104 acc:0 ppl:891.6992 Pre-training Discriminator...
Adversarial Training...
epoch:6 EmbeddingSimilarity:-0.7674406306576034 nll-test:3.5134504 tei:1.164234193166097 acc:0.0 ppl:891.6992 epoch:10 EmbeddingSimilarity:-0.5629098260707919 nll-test:3.579319 tei:3.0373429973920185 acc:0.03125 ppl:891.6992

******** Completed Training ********

Time Elapsed: 05mins:40secs

For below model training and test, results are discussed later. For more details, see Section **??**.

1.10.3 INFOGAN

The following commands are to run INFOGAN model on both oracle and real trainings.

NOTE: The real data essentially trained the model on the eapoe.txt data.

```
!python3 "main.py" -g infogan -t oracle -d $data -p $epoch
!python3 "main.py" -g infogan -t real -d $data -p $epoch
```

Oracle Training

[9]:
```
!python3 "main.py" -g infogan -t oracle -d $data -p $epoch
```

```
Unspecified Output Director Location: Defaulting to --> results/

**********************************************************************
Model Set:                          Infogan
Training Set:                       Oracle
Data Set:                           /content/CIS-700_clone/data/eapoe.txt
Epochs Set:                         5
Results Output Directory Set: results/

********************************************************************** Recording

Time Elapse...

********        Beginning       Training        ********
train_data      = results/generator_infogan_oracle.txt  test_data
= results/oracle_infogan_oracle.txt
Metrics Applied: nll-oracle, nll-test, EmbeddingSimilarity, tei, acc, ppl Pre-training Generator...
epoch:1                 nll-oracle:10.717546            nll-test:7.4416223
EmbeddingSimilarity:-0.21199913890316324    tei:1.9775988976160686    acc:0    ppl:10591.8633    Pre-
training Discriminator...
Adversarial Training...
epoch:6                 nll-oracle:10.527277            nll-test:7.247889
EmbeddingSimilarity:-0.211212207168559 tei:1.775283928712209       acc:0.46875 ppl:10591.8633
epoch:10                nll-oracle:10.445213            nll-test:7.282643
EmbeddingSimilarity:-0.20455168284975261    tei:3.8118852893511455                  acc:0.46875
ppl:10591.8633

******** Completed Training ********

Time Elapsed: 08mins:13secs
```

For below model training and test, results are discussed later. For more details, see Section **??**.

Real Training

[10]:
```
!python3 "main.py" -g infogan -t real -d $data -p $epoch
```

Unspecified Output Director Location: Defaulting to --> results/

****

Model Set: Infogan
Training Set: Real
Data Set: /content/CIS-700_clone/data/eapoe.txt
Epochs Set: 5
Results Output Directory Set: results/

********** Recording

Time Elapse...

******** Beginning Training ******** train_data = results/generator_infogan_real.txt test_data = results/oracle_infogan_real.txt
Metrics Applied: nll-test, EmbeddingSimilarity, tei, acc, ppl Pre-training Generator... epoch:1 EmbeddingSimilarity:-1.0819162003089347 nll-test:5.625535 tei:0.9641148765881856 acc:0 ppl:1068.0638 Pre-training Discriminator...
Adversarial Training...
epoch:6 EmbeddingSimilarity:-0.5316312323567908 nll-test:3.1113029 tei:1.190874715646108 acc:0.0 ppl:1068.0638 epoch:10 EmbeddingSimilarity:0.17387469852161172 nlltest:3.3262415 tei:3.2033821900685626 acc:0.0 ppl:1068.0638

******** Completed Training ********
Time Elapsed: 05mins:53secs

For below model training and test, results are discussed later. For more details, see Section **??**.

1.10.4 SeqGAN

The following commands are to run SeqGAN model on both oracle and real trainings. **NOTE:** The real data essentially trained the model on the eapoe.txt data.

!python3 "main.py" -g seqgan -t oracle -d $data -p $epoch
!python3 "main.py" -g seqgan -t real -d $data -p $epoch

Oracle Training

[11]:
```
!python3 "main.py" -g seqgan -t oracle -d $data -p $epoch
```

Unspecified Output Director Location: Defaulting to --> results/

****

Model Set: Seqgan

```
Training Set:                   Oracle
Data Set:                       /content/CIS-700_clone/data/eapoe.txt
Epochs Set:                     5
Results Output Directory Set: results/

********************************************************************** Recording Time

Elapse...

********        Beginning       Training        ********
train_data      = results/generator_seqgan_oracle.txt   test_data
= results/oracle_seqgan_oracle.txt
Metrics Applied: nll-oracle, nll-test, EmbeddingSimilarity, tei, acc, ppl Pre-training Generator... epoch:1
        nll-oracle:10.721059    nll-test:7.4408646
EmbeddingSimilarity:-0.21510319223100235    tei:1.9844633181889852 acc:0 ppl:11759.4696
Pre-training Discriminator... Adversarial Training...
epoch:6                 nll-oracle:10.541663            nll-test:7.2498097
EmbeddingSimilarity:-0.20923900373743434    tei:1.7679272850354513    acc:0.4375    ppl:11759.4696
epoch:10        nll-oracle:10.436457    nll-test:7.2877564      EmbeddingSimilarity:-
0.20577050242964026 tei:3.728414050738017 acc:0.625 ppl:11759.4696

******** Completed Training ********

Time Elapsed: 08mins:07secs
```

For below model training and test, results are discussed later. For more details, see Section **??**.

Real Training

[12]:
```
!python3 "main.py" -g seqgan -t real -d $data -p $epoch
```

```
Unspecified Output Director Location: Defaulting to --> results/

**********************************************************************
Model Set:                      Seqgan
Training Set:                   Real
Data Set:                       /content/CIS-700_clone/data/eapoe.txt
Epochs Set:                     5
Results Output Directory Set: results/

**********************************************************************

Recording Time Elapse...

******** Beginning Training ******** train_data = results/generator_seqgan_real.txt
        test_data       =
results/oracle_seqgan_real.txt
```

Metrics Applied: nll-test, EmbeddingSimilarity, tei, acc, ppl Pre-training Generator...
epoch:1 EmbeddingSimilarity:-1.215774718658185 nll-test:5.4708357 tei:0.9631365776062012
acc:0 ppl:770.7761 Pre-training Discriminator...
Adversarial Training...
epoch:6 EmbeddingSimilarity:-0.16652556276245745 nlltest:3.2102695
tei:1.1795339028040568 acc:0.0 ppl:770.7761 epoch:10 EmbeddingSimilarity:-
0.1827212026947554 nll-test:3.1191728 tei:3.198356052239736 acc:0.0625 ppl:770.7761

******** Completed Training ********

Time Elapsed: 05mins:52secs

For below model training and test, results are discussed later. For more details, see Section **??**.

1.10.5 TextGAN

The following commands are to run TextGAN model on both oracle and real trainings. **NOTE:** The real data essentially trained the model on the eapoe.txt data.

!python3 "main.py" -g textgan -t oracle -d $data -p $epoch
!python3 "main.py" -g textgan -t real -d $data -p $epoch

Oracle Training

[13]:
```
!python3 "main.py" -g textgan -t oracle -d $data -p $epoch
```

Unspecified Output Director Location: Defaulting to --> results/

****

Model Set: Textgan
Training Set: Oracle
Data Set: /content/CIS-700_clone/data/eapoe.txt
Epochs Set: 5
Results Output Directory Set: results/

********** Recording

Time Elapse...

******** Beginning Training ********

train_data = results/generator_textgan_oracle.txt test_data = results/oracle_textgan_oracle.txt
Metrics Applied: nll-oracle, nll-test, EmbeddingSimilarity, tei, acc, ppl Pre-training Generator...
epoch:1 nll-oracle:12.172784 nll-test:40.42925
EmbeddingSimilarity:-0.1965088573073615 tei:1.9016258319218953 acc:0 ppl:13132.1719
Pre-training Discriminator... Adversarial Training...
epoch:6 nll-oracle:11.784831 nll-test:36.225193

EmbeddingSimilarity:-0.31544909107702557 tei:1.5946348945299784 acc:1.0 ppl:13132.1719 epoch:10 nll-oracle:11.784829 nll-test:36.49654
EmbeddingSimilarity:-0.31743904604292605 tei:2.913949497540792 acc:1.0 ppl:13132.1719

******** Completed Training ********

Time Elapsed: 06mins:48secs

For below model training and test, results are discussed later. For more details, see Section **??**.

Real Training

[14]:
```
!python3 "main.py" -g textgan -t real -d $data -p $epoch
```

Unspecified Output Director Location: Defaulting to --> results/

****
Model Set: Textgan
Training Set: Real
Data Set: /content/CIS-700_clone/data/eapoe.txt
Epochs Set: 5
Results Output Directory Set: results/

********** Recording

Time Elapse...

******** Beginning Training ******** train_data = results/generator_textgan_real.txt test_data = results/oracle_textgan_real.txt
Metrics Applied: nll-test, EmbeddingSimilarity, tei, acc, ppl Pre-training Generator...
epoch:1 EmbeddingSimilarity:-0.4023692504412965 nll-test:32.949615 tei:0.9332664887110392 acc:0 ppl:903.9903 Pre-training Discriminator... Adversarial Training...
epoch:6 EmbeddingSimilarity:-0.4431972083946625 nll-test:33.8399 tei:1.19359845717748 acc:0.625 ppl:903.9903 epoch:10 EmbeddingSimilarity:-0.5257264523050709 nlltest:33.971855 tei:2.818093419075012 acc:0.71875 ppl:903.9903

******** Completed Training ********

Time Elapsed: 05mins:19secs

For below model training and test, results are discussed later. For more details, see Section **??**.

1.11 Results Comparision

Below sections are broken down into Data and Visual Representations of the results testing of models.

1.11.1 Data Representation

Below is a display of the best values for each training of each model run.

[15]:
```
import utils.visual as vis
vis.display_best_values()
```

********** Oracle Training:
********** Cgan nll-oracle: 9.9779 @epoch 10 nll-test: 7.8259 @epoch 6
EmbeddingSimilarity: -0.1296 @epoch 6 tei: 1.6517 @epoch 6 acc: 0.5938 @epoch 6 ppl: 3273.7598 @epoch 1
Dcgan nll-oracle: 10.553 @epoch 10 nll-test: 7.249 @epoch 6
EmbeddingSimilarity: -0.2076 @epoch 10 tei: 1.9748 @epoch 1 acc: 0.4688 @epoch 10 ppl: 6726.8167 @epoch 1

Infogan nll-oracle: 10.4452 @epoch 10 nll-test: 7.2479 @epoch 6
EmbeddingSimilarity: -0.2046 @epoch 10 tei: 1.7753 @epoch 6 acc: 0.4688 @epoch 6 ppl: 10591.8633 @epoch 1

Seqgan nll-oracle: 10.4365 @epoch 10 nll-test: 7.2498 @epoch 6
EmbeddingSimilarity: -0.2058 @epoch 10 tei: 1.7679 @epoch 6 acc: 0.625

@epoch 10 ppl: 11759.4696 @epoch 1

Textgan nll-oracle: 11.7848 @epoch 10 nll-test: 36.2252 @epoch 6
EmbeddingSimilarity: -0.1965 @epoch 1 tei: 1.5946 @epoch 6 acc: 1.0 @epoch 6 ppl: 13132.1719 @epoch 1

********** Real Training:
****
Cgan
EmbeddingSimilarity: -0.14 @epoch 10 nll-test: 6.0371 @epoch 6 tei: 0.9378 @epoch 1 acc: 0.9375 @epoch 6 ppl: 1020.2216 @epoch 1

Dcgan
EmbeddingSimilarity: -0.5629 @epoch 10 nll-test: 3.5135 @epoch 6 tei: 0.9595 @epoch 1 acc: 0.0312 @epoch 10 ppl: 891.6992 @epoch 1

Infogan
EmbeddingSimilarity: -0.1739 @epoch 10 nll-test: 3.1113 @epoch 6 tei: 0.9641 @epoch 1 acc: 0.0 @epoch 1 ppl: 1068.0638 @epoch 1

Seqgan

EmbeddingSimilarity: -0.1665 @epoch 6 nll-test: 3.1192 @epoch 10 tei: 0.9631 @epoch 1 acc: 0.0625 @epoch 10 ppl: 770.7761 @epoch 1

Textgan

EmbeddingSimilarity: -0.4024 @epoch 1 nll-test: 32.9496 @epoch 1 tei: 0.9333 @epoch 1 acc: 0.7188 @epoch 10 ppl: 903.9903 @epoch 1

1.11.2 Visual Representation

This sections provides a more visual representation of results and comparative data

[16]:
```
vis_display_synth_data()
```

Cgan Real Synth Data

0 pallid of and enchanted- can my sign above enchanted- sign floating for of of opened followed followed of gave all faintly laden
1 plainly token followed of mortal others but open tint open velvet thy lost followed followed floating sign and and of laden nodded
2 faintly to of ill above others tint mortal enchanted- mortal grew followed your of cried above nodded ? tint followed of pallid
3 silence dirges sign followed of fluttered- and of perched tint mortal of of good 't fiery followed followed ! startled stern !
4 only many for flitting and of stern followed above tint tint opened nothing followed followed ! of grim above can tint saintly

---

Dcgan Real Synth Data

0 there- fancy little disaster been 't you lore late shall store chamber uncertain " never window
1 ' ev'ry december so or bleak get nepenthe my , , get - quoth many lordly bleak fiend
2 many undaunted fancy one sainted saw taken lifted- " obeisance is reclining angels quaff " lordly lent on
3 curious in craven morrow ! me- books plume quaint guessing dirges many of and faintly before-
4 stately air much what straight stately dying hopes plume echo visitor let hath will beating i unto undaunted lore thee-

---

Infogan Real Synth Data

0 life clasp mortal chamber art flutter on gave
1 ; my peering 's
2 lore footfalls grew bird master
3 chamber spring bird
4 murmured methought gold tinkled window the plume dying

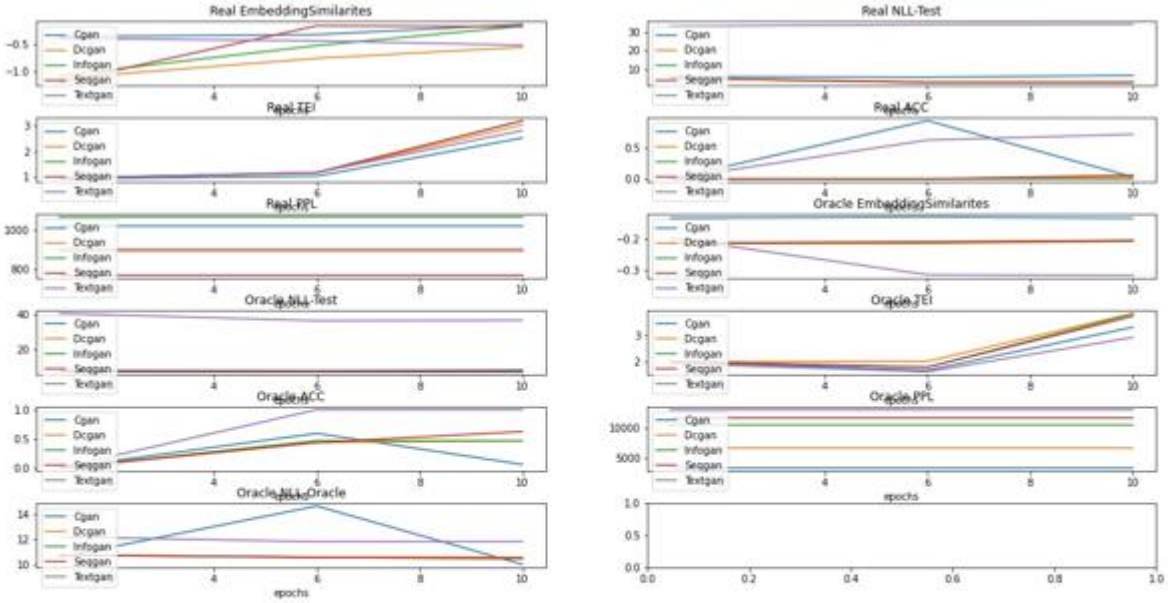
Seqgan Real Synth Data

0 bust at beast out some drawn .
1 `` shutter raven , many from gloated
2 lies - . nothing thee- token
3 black a taken , marvelled most the marvelled nothing
4 i was ;

Textgan Real Synth Data

0 loneliness gold of hath others gold of merely forgiveness fearing seen lenore- kind -- forgiveness of seeming our wondering merely velvet
1 forget bring before- devil bore bore hath all sainted of grim now silence aidenn bring in presently sent before- terrors kind seeming
2 all heard seat when both kind velvet this merely nightly presently beguiling all morrow rare devil devil pass morrow merely forgiveness lenore
3 merely december volume surcease now myself of curious this curious bring this sure bore beguiling merely forgiveness kind hath tempter devil depth
4 bore nodded flung gold feather seat just lenore- both curious his all spoke before- spoke spoke depth before- spoke december kind depth

[17]:
```
vis_display_metrics()
```



Observations

In terms of TEI, Cgan performed the best with Real TEI, while Textgan performed the best with Oracle TEI. It seems that Seqgan, Infogan, and Dcgan had similar performances for both Real and Oracle TEI, performing worse than both Cgan and Textgan in both metrics. For accuracy, Textgan performed the best across all epochs for Oracle accuracy, whereas Cgan performed better at lower epochs, but Textgan outperformed it in later epochs. One would assume Textgan would continue to eclipse Cgan for Real accuracy as the number of epochs increases. For perplexity (PPL), Seqgan had the best performance for real PPL, while Infogan had the worst. Meanwhile for ORacle PPL, Cgan had the best performance, while Textgan had the worst performance.