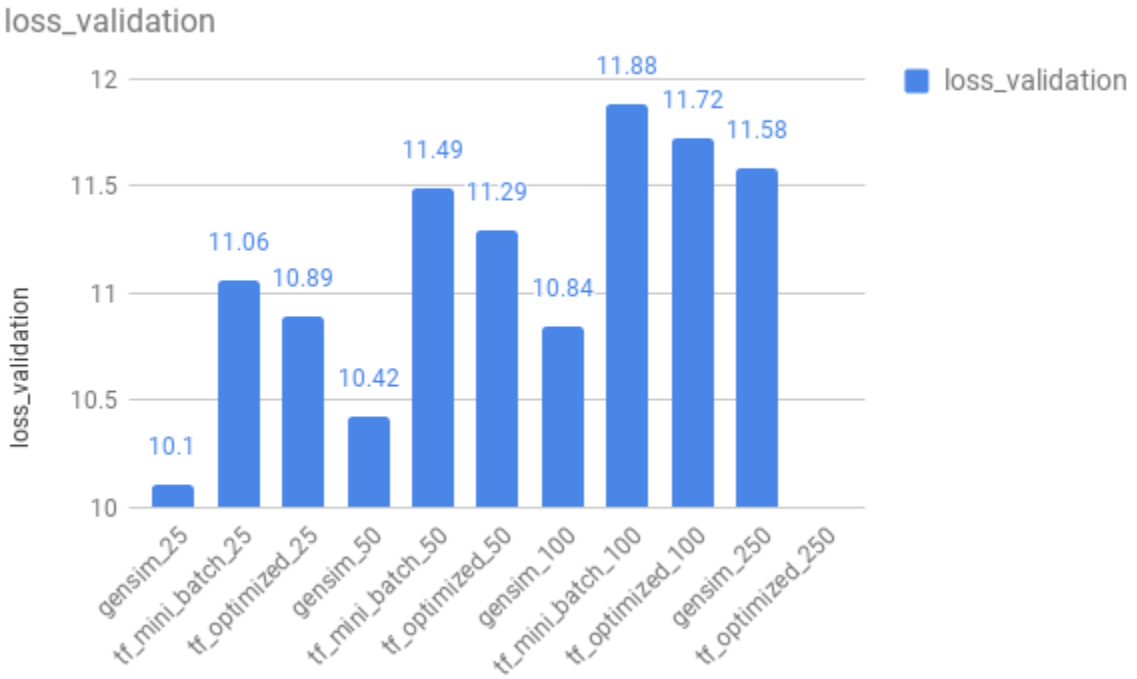
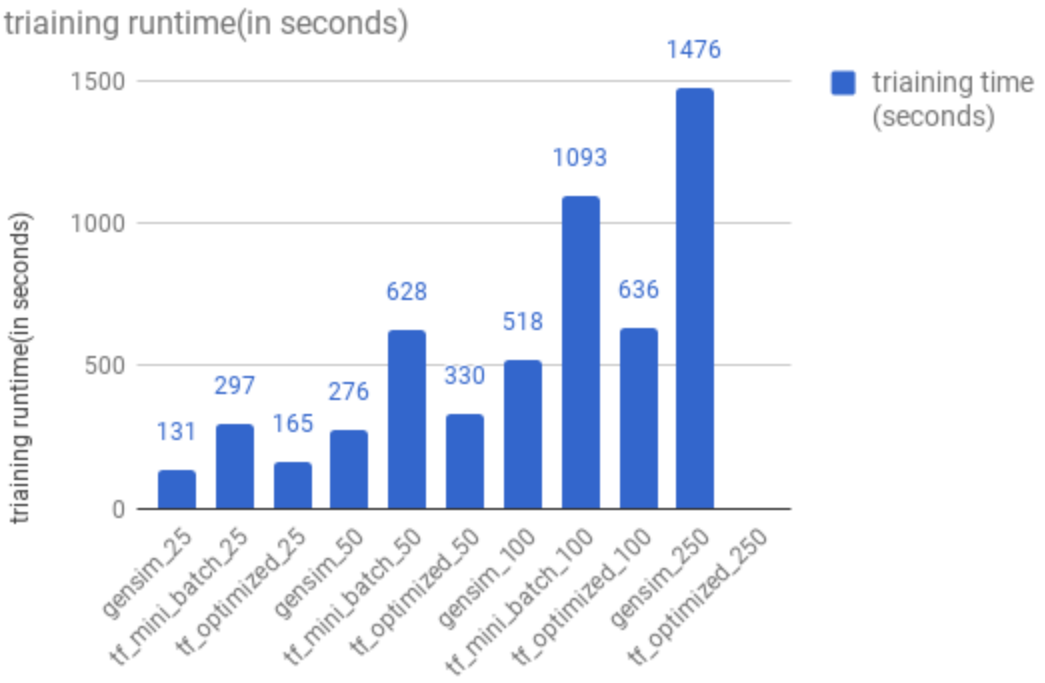
Gensim vs Tensorflow on Word2Vec

- **Result comparison on training time and validation loss between Tensorflow&gensim word2vec implementations:**

	gensim_25	tf_mini_batch_25	tf_optimized_25	gensim_50	tf_mini_batch_50	tf_optimized_50	gensim_100	tf_mini_batch_100
runtime(seconds)	131	297	165	276	628	330	518	1093
loss_validation	10.10	11.06	10.89	10.42	11.49	11.29	10.84	11.88

** The validation is did on **100** hold-out word pairs





- Other(not important) results:

	gensim_25	tf_mini_batch_25	tf_optimized_25	gensim_50	tf_mini_batch_50	tf_optimized_50	gensim_100	tf_mini_batch_100	tf_optimized_100	gensim_25
#total words	15153544	15153544	15153544	30249286	30249286	30249286	60451791	60451791	60451791	15103328
#unique words	724927	724927	724927	1063129	1063129	1063129	1582097	1582097	1582097	2643524
#vocabulary	62441	62441	62441	95880	95880	95880	146939	146939	146939	250330
words/sec	204036	around 90000	around 170000	203404	around 85000	around 180000	214461	around 85000	around 190000	204969
training_loss	NA	around 4	NA	NA	3.52	NA	NA	around 4	NA	NA
training_steps	NA	16142	16224	NA	33238	33417	NA	16117	68279	NA
#sentences	196480	196480	196480	391747	391747	391747	782650	782650	782650	1955801

- **Expriment settings:**

embedding_dim	100
window_size	5
#epoch	2
#threads	8
min_count	20
downsampling rate	1e-03
#negative samples	5
initial learning rate	0.025
mini-batch size(if applicable)	10000

- Step-by-Step documentation(Please refer the enclosed ReadMe.pdf for better formatting)

Step 0: Prepare the training data

Run :

```
python gen_tf_data.py --input_path ~/Data/user-history/sample_150/
```

Where the `--input_path` is the folder path which contains `user-logs`
It will generate a data file named `text8.txt`. Rename it and put it in some folder. The path to this file will be used later on.

Step 1: Build custom operator in C++

on MacOS:

```
TF_INC=$(python -c 'import tensorflow as tf; print(tf.sysconfig.get_include())')
g++ -std=c++11 -undefined dynamic_lookup -O3 -shared word2vec_ops.cc word2vec_kernels.cc -o
word2vec_ops.so -fPIC -I $TF_INC -O2 -D_GLIBCXX_USE_CXX11_ABI=0
```

on Linux:

```
TF_INC=$(python -c 'import tensorflow as tf; print(tf.sysconfig.get_include())')
g++ -std=c++11 -O3 -shared word2vec_ops.cc word2vec_kernels.cc -o word2vec_ops.so -fPIC -I $TF_INC
-O2 -D_GLIBCXX_USE_CXX11_ABI=0
```

Step 2: Pass arguments to run tensorflow word2vec

Arguments:

- must provide:
  `--save_path`
  Directory to write the model.
  `--train_data`
  Training data, Unziped single txt file.
- optional:
  `--num_corpus`
  Number of User-log gzip files for training. default is 250. No need to specify in runtime.
  `--embedding_size`
  The embedding dimension size. Default 100.
  `--epochs_to_train`
  Number of epochs to train. Each epoch processes the training data once completely. Default 2.
  `--learning_rate`
  Initial learning rate. Default 0.025.
  `--num_neg_samples`
  Negative samples per training example. Default 5
  `--batch_size`
  Numbers of training examples each step processes (no minibatching). Default 10000
  `--concurrent_steps`

The number of concurrent training steps(number of threads/workers). Default 8
`--window_size`
The number of words to predict to the left and right of the target word. Default 5
`--min_count`
The minimum number of word occurrences for it to be included in the vocabulary. Default 20
`--subsample`
Subsample threshold for word occurrence. Words that appear with higher frequency will be randomly down-sampled. Set to 0 to disable. Default 1e-3

Step 3: Run the test

for each of the folowing settings, the code will save the trained `embeddings` and `output weights` in `--save_path`. Which can be used for evaluation later on.

word2vec with minibatch = 10000

```
python word2vec.py  --num_corpus 25 --train_data=text8_25.txt
--eval_data=/Users/polybahn/Desktop/w2v_test/test-text.txt
--save_path=/Users/polybahn/Outputs/word2vec/tf_sample --epochs_to_train 2

python word2vec.py --num_corpus 50 --train_data=text8_50.txt
--eval_data=/Users/polybahn/Desktop/w2v_test/test-text.txt
--save_path=/Users/polybahn/Outputs/word2vec/tf_sample --epochs_to_train 2

python word2vec.py --num_corpus 100  --train_data=text8_100.txt
--eval_data=/Users/polybahn/Desktop/w2v_test/test-text.txt
--save_path=/Users/polybahn/Outputs/word2vec/tf_sample --epochs_to_train 2

python word2vec.py --num_corpus 250  --train_data=text8_250.txt
--eval_data=/Users/polybahn/Desktop/w2v_test/test-text.txt
--save_path=/Users/polybahn/Outputs/word2vec/tf_sample --epochs_to_train 2
```

word2vec without minibatch

```
python word2vec_optimized.py --num_corpus 25  --train_data=text8_25.txt
--eval_data=/Users/polybahn/Desktop/w2v_test/test-text.txt
--save_path=/Users/polybahn/Outputs/word2vec/tf_op_sample --epochs_to_train 2

python word2vec_optimized.py --num_corpus 50  --train_data=text8_50.txt
--eval_data=/Users/polybahn/Desktop/w2v_test/test-text.txt
--save_path=/Users/polybahn/Outputs/word2vec/tf_op_sample --epochs_to_train 2

python word2vec_optimized.py --num_corpus 100  --train_data=text8_100.txt
--eval_data=/Users/polybahn/Desktop/w2v_test/test-text.txt
--save_path=/Users/polybahn/Outputs/word2vec/tf_op_sample --epochs_to_train 2

python word2vec_optimized.py --num_corpus 250  --train_data=text8_250.txt
--eval_data=/Users/polybahn/Desktop/w2v_test/test-text.txt
--save_path=/Users/polybahn/Outputs/word2vec/tf_op_sample --epochs_to_train 2
```

word2vec using gensim

```
python run_gensim.py --output_path ~/Outputs/word2vec/gensim_sample/ --corpus_size 25 --num_epochs 2
--num_workers 8 --num_dims 100 --min_count 20 --window_size 5 --input_path
~/Data/user-history/sample_25/

python run_gensim.py --output_path ~/Outputs/word2vec/gensim_sample/ --corpus_size 50 --num_epochs 2
--num_workers 8 --num_dims 100 --min_count 20 --window_size 5 --input_path
~/Data/user-history/sample_50/

python run_gensim.py --output_path ~/Outputs/word2vec/gensim_sample/ --corpus_size 100 --num_epochs 2
--num_workers 8 --num_dims 100 --min_count 20 --window_size 5 --input_path
~/Data/user-history/sample_100/

python run_gensim.py --output_path ~/Outputs/word2vec/gensim_sample/ --corpus_size 250 --num_epochs 2
--num_workers 8 --num_dims 100 --min_count 20 --window_size 5 --input_path
~/Data/user-history/sample_250/
```

Step 4: Evaluation. Please refer to `Word_Embedding_Eval.ipynb`

The Eveluation reads the embedding weights and output weights, then calculate the average standard validation loss on the holdout dataset.
Default runs on 100 pairs.

Unsolved issues:

- In `word2vec_kernels.cc` `Line 183`: Read all data into a single `string` type variable, which will exceeds the limit. Need to change to some line by line reader.
- Need to look how to read from S3 in C++, then unpack, then read line by line.