pa - w2v mono training 1

June 2, 2019

1 Gensim Training Experiments

- **Machines:**
  - HEIA-FR GPU-2 (32 cpu dual threaded)
  - CPU Monster at HEIA-FR (48 cpu single threaded)
- **Dataset:**
  - wikipedia english dump from 2019-03-19 (16GB)
  - wikipedia english dump from 2019-04-09 (16GB)
- **Dictionary:**
  - lemmatized dictionary(16MB)
  - unlemmatized dictionary(16MB)

1.1 What's going on

- Training a Word2Vec on the full wikipedia english dataset using its pre-extracted lemmatized and unlemmatized dictionary.

In [1]:
```
# Word2Vec settings
import multiprocessing

#w2v_w2v_sentences=None
#w2v_corpus_file=None
w2v_size=300 # (default: 100)
#w2v_alpha=0.025
w2v_window=10 # (default: 5)
w2v_min_count=1 # (default: 5)
#w2v_max_vocab_size=None
#w2v_sample=0.001
#w2v_seed=1
w2v_workers=4 # (default: 3) # multiprocessing.cpu_count()
#w2v_min_alpha=0.0001
w2v_sg=0 # if sg=0 CBOW is used (default); if sg=1 skip-gram is used
#w2v_hs=0
#w2v_negative=5
```

```
        #w2v_ns_exponent=0.75
        #w2v_cbow_mean=1
        #w2v_hashfxn=<built-in function hash>
        w2v_iter=5 # (default: 5)
        #w2v_null_word=0
        #w2v_trim_rule=None
        #w2v_sorted_vocab=1
        w2v_batch_words=10000 # (default: 10000)
        #w2v_compute_loss=False
        #w2v_callbacks=()
        #w2v_max_final_vocab=None
```

In [2]:
```
        # General settings
        lemmatization = False
        run_corpus = "wiki"
        run_lang = "en"
        run_date = "190409"
        run_log_prefix = "train"

        run_model_dir = "models/"
        run_dict_dir = "dictionaries/"
        run_datasets_dir = "datasets/"
        run_log_dir = "logs/"
```

In [3]:
```
        run_w2v_algo = "cbow" if w2v_sg==0 else "sg"

        run_options = "s"+str(w2v_size)+"-w"+str(w2v_window)+"-mc"+str(w2v_min_count)+"-bw"+str
        print(run_options)

        run_base_name = run_corpus+"-"+run_lang+"-"+run_date # wiki-en-190409
        run_model_name = run_model_dir+run_base_name+"-"+run_options

        run_dict_name = run_dict_dir+run_base_name+"-dict"
        run_dataset_name = run_datasets_dir+run_base_name+"-latest-pages-articles.xml.bz2"
        run_log_name = run_log_dir+run_log_prefix+"-"+run_base_name+"-"+run_options

        run_lem = "-lem" if lemmatization else "-unlem"

        run_model_name += run_lem+".model"
        run_dict_name += run_lem+".txt.bz2"
        run_log_name += run_lem+".log"

        print(run_model_name)
        print(run_dict_name)
        print(run_dataset_name)
        print(run_log_name)
```

```
s300-w10-mc1-bw10000-cbow-i5-c4
models/wiki-en-190409-s300-w10-mc1-bw10000-cbow-i5-c4-unlem.model
```

```
dictionaries/wiki-en-190409-dict-unlem.txt.bz2
datasets/wiki-en-190409-latest-pages-articles.xml.bz2
logs/train-wiki-en-190409-s300-w10-mc1-bw10000-cbow-i5-c4-unlem.log
```

In [4]:
```
# Start logging process at root level
import logging
logging.basicConfig(filename=run_log_name, format='%(asctime)s : %(levelname)s : %(mess
#logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging
logging.root.setLevel(level=logging.INFO)
```

In [ ]:

In [5]:
```
# Load dictionary from file
from gensim.corpora import Dictionary
dictionary = Dictionary.load_from_text(run_dict_name)
```

In [ ]:
```
# Build WikiCorpus based on the dictionary
from gensim.corpora import WikiCorpus

wc_fname=run_dataset_name
#wc_processes=None
wc_lemmatize=lemmatization
wc_dictionary=dictionary
#wc_filter_namespaces=('0', )
#wc_tokenizer_func=<function tokenize>
#wc_article_min_tokens=50
#wc_token_min_len=2
#wc_token_max_len=15
#wc_lower=True
#wc_filter_articles=None

wiki = WikiCorpus(fname=wc_fname, dictionary=wc_dictionary,lemmatize=wc_lemmatize)
```

In [ ]:
```
# Initialize simple sentence iterator required for the Word2Vec model
# Trying to bypass memory errors

if lemmatization:
    class SentencesIterator:
        def __init__(self, wiki):
            self.wiki = wiki

        def __iter__(self):
            for sentence in self.wiki.get_texts():
                yield list(map(lambda x: x.decode('utf-8'), sentence))
                #yield gensim.utils.simple_preprocess(line)

else:
    class SentencesIterator:
```

```
            def __init__(self, wiki):
                self.wiki = wiki

            def __iter__(self):
                for sentence in self.wiki.get_texts():
                    yield list(map(lambda x: x.encode('utf-8').decode('utf-8'), sentence))

        sentences = SentencesIterator(wiki)
```

In [ ]:
```
        # Train model
        from gensim.models import Word2Vec

        print("Running with: " + str(w2v_workers) + " cores")

        model = Word2Vec(sentences=sentences,
                         size=w2v_size,
                         window=w2v_window,
                         min_count=w2v_min_count,
                         workers=w2v_workers,
                         sg=w2v_sg,
                         iter=w2v_iter
                        )
        model.save(run_model_name)

        del model
        del wiki
        del sentences
        del dictionary
```

```
Running with: 4 cores
```

In [ ]: