Tom 1 Topic Mode…

READY

```
%pyspark

# PySpark CommonCrawl Topic Modelling
# Tom V / Paul J - 8/11/2017

# SET THE spark.driver.maxResultSize PROPERTY TO 3G

import boto
from boto.s3.key import Key
from gzipstream import GzipStreamFile
from pyspark.sql.types import *
import warc
import ujson as json
from urlparse import urlparse
from langdetect import detect_langs
import pycld2 as cld2

wetlist = sc.textFile("s3://commoncrawl/crawl-data/CC-MAIN-2017-04/wet.paths.gz")
wetlist.cache()

def unpack(uri):
    conn = boto.connect_s3(anon=True, host='s3.amazonaws.com')
    bucket = conn.get_bucket('commoncrawl')
    key_ = Key(bucket, uri)
    file_ = warc.WARCFile(fileobj=GzipStreamFile(key_))
    return file_

def detect(x):
    try:
        return detect_langs(x)[0].lang # Maybe we can get away with looking at less cha
    except Exception as e:
        return None

def detect2(x):
    try:
        isReliable, textBytesFound, details = cld2.detect(x)
        return details[0][1]
    except Exception as e:
        print(e)
        return None

def process_wet(id_, iterator):
    for uri in iterator:
        file = unpack(uri)
        for record in file: # Approx 53k web pages per WET file
            try:
                #url = record.rec_headers.get_header('WARC-Target-URI')
                #yield record, record.content_stream().read().decode('utf-8')
                url = record.url
                domain = None if not url else urlparse(url).netloc
```

```
                text = record.payload.read().decode('utf-8') #.limit(100) # TODO: Limit
                lang = detect2(text[:300]) # TODO: Language detection is what appears t
                yield domain, url, text, lang
            except Exception as e:
                yield e

def process_wet_simple(id_, iterator):
    count=0
    for uri in iterator:
        file = unpack(uri)
        for record in file:
            try:
                count=count+1
                # TODO: Output total size of pages, rather than number of pages
                # Histogram.
            except Exception as e:
                pass
        #print(count)
```

```
%pyspark
detect2("this is a test")
```

READY

```
'en'
```

```
%pyspark

# PARAMETER - number of input files
nfiles = 2048

# PARAMETER - slices / partitions of input
files = sc.parallelize(wetlist.take(nfiles), numSlices=nfiles)

# TODO: Make this use more than one CPU!
print(files.getNumPartitions())
#files.mapPartitionsWithIndex(process_wet_simple).collect()
rdd=files.mapPartitionsWithIndex(process_wet)

print(str(rdd))
docs = rdd.toDF(["domain", "url", "text","lang"]) #  "lang"
#docs.cache()
docs.count()
```

READY

```
2048
PythonRDD[71] at RDD at PythonRDD.scala:48
109857736
```

```
%pyspark

docs_en = docs.filter(docs.lang == 'en')
#docs_en = docs
```

READY

```
# PARAMETER - possibly set partitions?
#docs_en = docs_en.repartition(nfiles)
#docs_en.rdd.getNumPartitions()
#docs_en.sample(True,0.01).groupBy('lang').count().toPandas()
docs_en.count()
```

43818581

```
%pyspark
```

READY

```
stopwords_english = ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you'
    'herself', 'it', 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves',
    'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing'
    'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before'
    'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how',
    'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just'
    'doesn', 'hadn', 'hasn', 'haven', 'isn', 'ma', 'mightn', 'mustn', 'needn', 'shan',

from pyspark.ml import Pipeline
from pyspark.ml.feature import RegexTokenizer, CountVectorizer, StopWordsRemover

# PARAMETER - regex tokenization
tokenizer = RegexTokenizer(inputCol="text", outputCol="words", pattern="\\p{L}{3,}", ga
stopwordRemover = StopWordsRemover(inputCol="words", outputCol="filtered",stopWords=sto

# PARAMETER - vocab size, min and max doc frequency
cv = CountVectorizer(inputCol="filtered", outputCol="vec",vocabSize=50000, minDF=50) #

# TODO: Investigate why this hangs with 256 files
pipeline = Pipeline(stages=[tokenizer, stopwordRemover, cv])
model = pipeline.fit(docs_en)
vecs = model.transform(docs_en).drop('text').drop('words').drop('filtered')
vecs.cache()
vecs.count()

# Took 42 min 51 sec on m4.16xlarge with 1024 files
```

43818581

```
%pyspark
sc.getConf().get('spark.driver.maxResultSize')
```

READY

u'16g'

```
%pyspark
model.save('s3://billsdata.net/CommonCrawl/topic_model_%d_files/textmodel' % nfiles)
vecs.write.parquet('s3://billsdata.net/CommonCrawl/topic_model_%d_files/cc_english_page
```

READY

Skip to here to restart from saved vectors

FINISHED

```
%pyspark
from pyspark.ml.pipeline import PipelineModel

nfiles = 2048

model2 = PipelineModel.load('s3://billsdata.net/CommonCrawl/topic_model_%d_files/textmo
vecs2 = spark.read.parquet('s3://billsdata.net/CommonCrawl/topic_model_%d_files/cc_engl

vecs2.show(10)
```

```
+--------------------+--------------------+----+--------------------+
|              domain|                 url|lang|                 vec|
+--------------------+--------------------+----+--------------------+
|                null|                null|  en|(50000,[60,79,92,...|
|1.cricwaves-hrd.a...|http://1.cricwave...|  en|(50000,[0,1,2,3,5...|
|       1015store.com|http://1015store....|  en|(50000,[0,6,7,23,...|
|101bestandroidapp...|http://101bestand...|  en|(50000,[0,2,4,5,6...|
|101bestandroidapp...|http://101bestand...|  en|(50000,[0,2,4,5,7...|
|1027jackfm.iheart...|http://1027jackfm...|  en|(50000,[0,4,5,6,9...|
|     1057thehawk.com|http://1057thehaw...|  en|(50000,[0,1,2,3,5...|
|     1063thebuzz.com|http://1063thebuz...|  en|(50000,[0,2,3,5,6...|
|     1063thebuzz.com|http://1063thebuz...|  en|(50000,[0,2,3,5,6...|
|  1190kex.iheart.com|http://1190kex.ih...|  en|(50000,[0,4,5,6,7...|
+--------------------+--------------------+----+--------------------+
only showing top 10 rows
```

```
%pyspark                                                                    FINISHED
vecs2.cache()
```

```
DataFrame[domain: string, url: string, lang: string, vec: vector]
```

```
%pyspark                                                                    FINISHED

# Run the topic modelling

from pyspark.ml.clustering import LDA
#inputCol="vec", outputCol="ldaVec", k=3, optimizer="online"

# Fix java memory errors, perhaps using:
# spark.driver.memory 256g - DIDN'T WORK
# or by reducing vocabSize from 100k to 20k - WORKS!

#With 128:
#    Py4JJavaError: An error occurred while calling o375.fit.
#: org.apache.spark.SparkException: Job 49 cancelled because SparkContext was shut down

lda = LDA(k=100, maxIter=10, featuresCol="vec") # Reduced maxIter from 100 to 50
ldaModel = lda.fit(vecs2)
```

```
%pyspark                                                                    FINISHED
```

```
# Save the models
ldaModel.save('s3://billsdata.net/CommonCrawl/topic_model_%d_files/ldamodel' % nfiles)
#pipeline.save('s3://billsdata.net/CommonCrawl/topic_model_%d_files/textpipeline' % nfi
```

```
%pyspark
```

FINISHED

```
# Get topic vectors for index pages (estimate of topic vec per domain)
vecs_index = vecs2.filter("url LIKE '%index.html'")
results = ldaModel.transform(vecs_index)

# Drop text cols
results2=results.drop('text').drop('words').drop('filtered')

# Save domain topic vecs
results2.write.parquet('s3://billsdata.net/CommonCrawl/topic_model_%d_files/cc_index_pa
```

```
%pyspark
```

FINISHED

```
# Create a dataset containing just the host, url and top 3 topic labels & scores
import pandas as pd
import numpy as np
topicIndices = ldaModel.describeTopics(maxTermsPerTopic = 5).collect()
vocab = model2.stages[2].vocabulary

topic_labels = []
for i, (topic, terms, termWeights) in enumerate(topicIndices):
    topwords = pd.Series(dict(zip([vocab[t] for t in terms], termWeights))).sort_values
    topic_labels.append('_'.join(topwords.index.values))

topic_labels = np.array(topic_labels)

def topTopics(x):
    labels = topic_labels[np.argsort(x.topicDistribution)[::-1][:3]]
    scores = np.sort(x.topicDistribution)[::-1][:3]
    return (x.domain, x.url, str(labels[0]), float(scores[0]), str(labels[1]), float(sc

results3 = results2.rdd.map(topTopics)
results3 = results3.toDF(["host", "url", "topic1", "score1", "topic2", "score2", "topic

results3.write.parquet('s3://billsdata.net/CommonCrawl/topic_model_%d_files/cc_index_pa
results3.show()
```

```
+--------------------+--------------------+--------------------+--------------------+----
----------------+--------------------+--------------------+--------------------+
|                host|                 url|              topic1|              score1|
topic2|              score2|              topic3|              score3|
+--------------------+--------------------+--------------------+--------------------+----
----------------+--------------------+--------------------+--------------------+
|    arabiansites.com|http://arabiansit...|dell_poweredge_ne...| 0.2821541632452619|hote
ls_hotel_reso...| 0.25533752747683996|new_john_lhblk_st...| 0.09580504989083381|
|      boysgayxxx.com|http://boysgayxxx...|porn_sex_tube_vid...| 0.9567347637439242|part
s_products_eq...|0.016390526468860557|wedding_info_indi...|0.0106102403034195 64|
|      boysgayxxx.com|http://boysgayxxx...|porn_sex_tube_vid...| 0.9706813926413853|wedd
ing_info_indi...|0.012924001701882118|width_font_color_...|0.012084585285042889|
|        cfbstats.com|http://cfbstats.c...|sports_news_baske...| 0.7573308264452827|news
_new_sports_g...|  0.2266831374043146|page_wiki_add_com...|1.661645362495028...|
|     dailysudoku.com|http://dailysudok...| jan_dec_nov_oct_mar| 0.5286669139625906|janu
ary_december_...|  0.2774674758221686|table_winchester_...|    0.13024862559943|
|  distro.ibiblio.org|http://distro.ibi...|files_linux_photo...| 0.4315107404072598|com_
game_games_se...|  0.35795739451731 9|buy_dictionary_mu...| 0.09397649357302677|
```

```
%pyspark
```

READY