Share 0 More Next Blog» Create Blog Sign In

Big Data Bloggin

Wednesday, August 22, 2012

trending topics in Hive

<WARNING> I normally try to keep the Big Data discussions in this blog accessible to non-geeks, this is anything but. There is a lot of nasty HQL, terminology, etc. I promise the next post will get back on track. </WARNING>

So I recently unearthed a bunch of discussions and stuck them in my Hive cluster (I'm adding about 4K a day). I loved the n-gram examples in the Hadoop the Definitive Guide (which I loaned out and was never returned!) and in the Amazon EMR docs. Now I had a nice data set to work with!

Unfortunately all the hive examples I've found either leveraged a publicly available n-gram set, or didn't really do anything with it. Getting the n-grams from my hive table is easy enough with a lifted hive command and my table structure (table:discussions, column:description):

```
hive> SELECT explode(context_ngrams(sentences(lower(description)), array(null), 25))
AS word_map FROM discussions;
```

But unfortunately the results come back in a hideously ugly structure and full of words that don't tell us much. I had some work to do ...

```
{"ngram":["the"],"estfrequency":82998.0}
{"ngram":["to"],"estfrequency":53626.0}
{"ngram":["and"],"estfrequency":51487.0}
{"ngram":["of"],"estfrequency":41579.0}
{"ngram":["a"],"estfrequency":37644.0}
{"ngram":["in"],"estfrequency":29572.0}
{"ngram":["i"],"estfrequency":27989.0}
{"ngram":["is"],"estfrequency":26268.0}
{"ngram":["that"],"estfrequency":23225.0}
{"ngram":["for"],"estfrequency":17557.0}
{"ngram":["be"],"estfrequency":14100.0}
{"ngram":["are"],"estfrequency":12911.0}
{"ngram":["with"],"estfrequency":12660.0}
{"ngram":["it"],"estfrequency":12618.0}
{"ngram":["this"],"estfrequency":12091.0}
{"ngram":["as"],"estfrequency":12023.0}
{"ngram":["have"],"estfrequency":11179.0}
{"ngram":["my"],"estfrequency":10446.0}
{"ngram":["or"],"estfrequency":10011.0}
{"ngram":["on"],"estfrequency":9602.0}
{"ngram":["you"],"estfrequency":9460.0}
{"ngram":["not"],"estfrequency":8521.0}
{"ngram":["they"],"estfrequency":7306.0}
{"ngram":["would"],"estfrequency":7093.0}
{"ngram":["can"],"estfrequency":7087.0}
```

It was clear I needed some sort of whitelist to find anything interesting (83K "the's" don't tell me much), so I borrowed this one put it into a new Hive table.

```
hadoop fs -mkdir /temp8/
hadoop fs -cp s3://XXX/stopwords.txt /temp8/.
hive -e  'drop table stop_words '
hive -e  'create table stop_words (word string) row format delimited fields terminated by
"\t";
load data inpath "/temp8/" overwrite into table stop_words;'
```

Blog Archive

- ► 2013 (4)
- ▼ 2012 (10)
  - ► December (1)
  - ► November (1)
  - ► October (1)
  - ► September (1)
  - ▼ August (2)
    - trending topics in Hive
    - the death of BI
  - ► July (1)
  - ► June (2)
  - ► May (1)

About Me

trippytom

I enjoy the search for technology which actually simplifies my life, most things West African, and jokes.

View my complete profile

Now I needed to get the data into a structure I could use. After playing around, I was able to get the returned struct in a Hive table. This was important because there are a lot of limitations on what you can do with Hive UDTF's (explode, etc), plus it takes a minute or so to run the n-grams query. Trust me, it sucks.

```
hive -e  'drop table trending_words_struct;'
hive -e  'create table trending_words_struct (NEW_ITEM
ARRAY<STRUCT<ngram:array<string>, estfrequency:double>>);
INSERT OVERWRITE TABLE trending_words_struct
SELECT context_ngrams(sentences(lower(description)), array(null), 1000) as
word_map FROM discussions;'
```

Next, I can unlock this mess of a structure and put it into something easier to query.

```
hive -e  'drop table trending_words;'
hive -e  'create table trending_words (ngram string, estfrequency double);'
hive -e 'INSERT OVERWRITE TABLE trending_words select X.ngram[0],
X.estfrequency  from (select explode(new_item) as X from trending_words_struct) Z;';
```

Finally I can match against the whitelist:

```
select tw.ngram, tw.estfrequency from trending_words tw LEFT OUTER JOIN
stop_words sw ON (tw.ngram = sw.word) WHERE sw.word is NULL order by
tw.estfrequency DESC;
```

… And get a list of non-trival words from my discussions! Granted, I need to add stuff like can/will/etc to my whitelist, change to daily inspection, etc. But you get the idea.

```
can 5096.0
will 4145.0
one 3018.0
also 2953.0
may 2397.0
children 2187.0
think 2088.0
people 2025.0
time 2007.0
use 1942.0
help 1912.0
behavior 1838.0
information 1755.0
research 1748.0
like 1673.0
work 1654.0
many 1650.0
http 1590.0
child 1552.0
make 1542.0
health 1538.0
2010 1518.0
well 1482.0
new 1481.0
used 1480.0
good 1439.0
different 1432.0
2012 1410.0
```

The key technical breakthroughs were:

#1 Deciding to just dump the results in their nasty structure into a Hive table
ARRAY<STRUCT<ngram:array<string>, estfrequency:double>>

#2 Figuring out how to get those results into a usable table (that Z was a killer!)

```
select X.ngram[0], X.estfrequency from (select explode(new_item) as X from
trending_words_struct) Z;
```

#3 getting the outer JOIN right

```
select tw.ngram, tw.estfrequency from trending_words tw LEFT OUTER JOIN
stop_words sw ON (tw.ngram = sw.word) WHERE sw.word is NULL order by
tw.estfrequency DESC;
```

Sadly, there was a lot of trial and error. Dead ends were frequent, a Hive bug was encountered, etc. It took me days to get this figured out, and I wanted to document the process in hopes others benefit from my mistakes.

The cool thing is this treatment will work for just about anything. Have at it!

Posted by trippytom at 9:54 AM

Recommend this on Google

No comments:

Post a Comment

Enter your comment...

Comment as: Google Account

Publish Preview

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)

Ethereal template. Powered by Blogger.