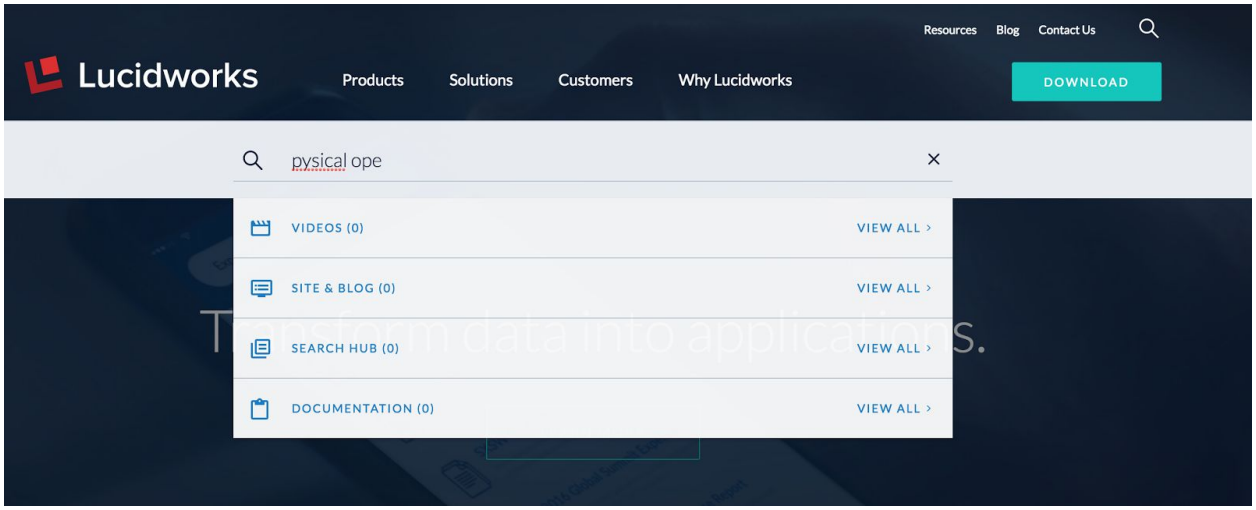
Autocorrect-Typeahead Suggestions

July 13, 2017

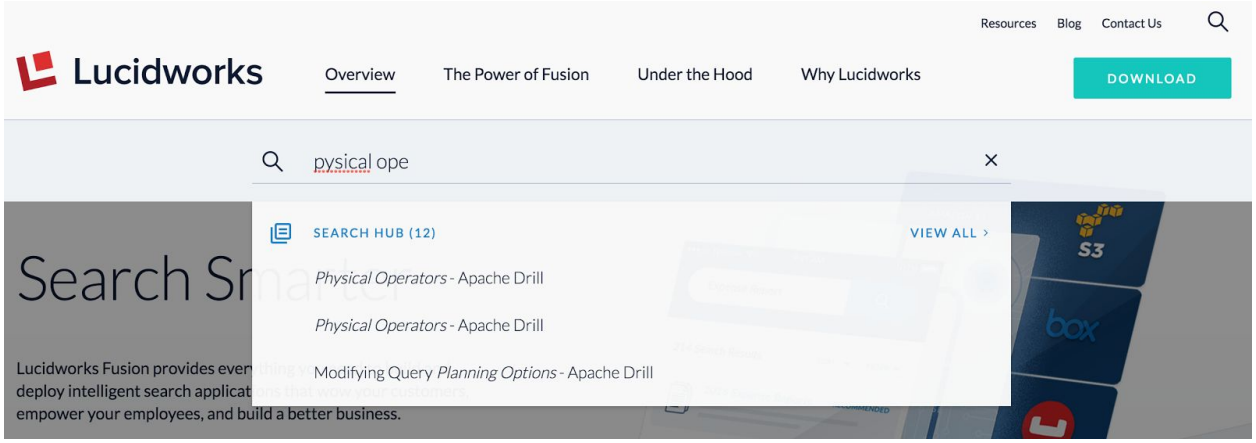
Concept:

The autocorrect feature is based on top of the typeahead feature work done by Yann. Autocorrect has been designed and implemented in a way that requires minimum changes in the front-end of the website. Literally one line change in ***ajax.php*** where the 'typeahead' search request is handled. This has made it easy to incorporate the feature easily in the site.

Website before autocorrect:



Website after autocorrect:



At the front end the UI only makes a FUSION API call and receives the suggestions while the FUSION Pipelines handle the spelling mistakes.

Initial plan was to use Levenshtein distance to compute the closest correctly spelled query available in our collection. However, as the dataset gets larger this approach will have issues at runtime and would result in very poor performance in terms of time. Thus we use a second approach.

Working principle:

1. Create ngram, edge-ngram indexes on the titles in the lucidfind collection. Add these fields in the solr schema, a datasource to index the titles in the lucidfind collection using the searchhub's bootstrap.py script.

2. Add ngram, edge-ngram query analyzers in solr. This is the key part of autocorrect. To understand how this works consider the following example:
   For e.g. lets consider bigram and trigram indexing.
   `pysiscal` does not match directly with any of the indexes, if we break down the query itself into bi/tri grams we get ***[py,ys,si,is,sc,ca,al,pys,ysi,sis,isc,sca,cal]*** **11/13** chunks match with the index of `physical operators` and it will thus show the suggestions.

3. Ngrams and edge-ngrams are important here because these indexes have been created on the titles and stored in a solr collection. The incoming query is broken down into chunks like the above example and a simple solr query call is made. All the matching work happens at the backend and the results are returned to the front end. Since, the indexes are precomputed and stored this approach is a lot faster and gives good runtime performance over computing any distances at runtime.

Execution Instructions:

Pre-req: Have the lucidfind collection ready with data indexed into it

1. `git clone -b typeahead-autocorrect https://github.com/lucidworks/searchhub.git`
2. `Set fusion home in gradle.properties file`
3. `./gradlew install`
4. `Set fusion username password in ./python/config.py`
5. `~searchhub$source venv/bin/activate`
6. `~searchhub$cd python`
7. `~python$../venv/bin/python bootstrap.py --setup_autocorrect_typeahead`
8. `If the above command runs successfully, it will have created the autocorrect-typeahead collection, autocorrect-typeahead query pipeline, and a datasource to index titles.`

9. Select the autocorrect-typeahead collection in the UI and crawl the datasource and let it run to completion. On success you should be able to search using the query-workbench. Select the autocorrect-typeahead collection, load the autocorrect-typeahead pipeline from the UI.
10. Try out some queries like `phsic` and hit enter. You should see some results. Those are your autocorrect-typeahead suggestions.

Watch the results on the UI:

1. Either use the sample_ui.html file to see the results or you can modify the **ajax.php** file and change the pipeline and collection there.
2. Refer the website github readme for instructions on how to set it up.

Change this line to point to your local shub instance. Do not use localhost

Change these line. Search for the commented out line (669 in this case) and replace it with the line (670 in this case). Restart the website and navigate to ***new.lucidworks.com*** and try out some queries.