Lucidworks

# Chatting with Solr

Erik Hatcher, Lucidworks
erik.hatcher@lucidworks.com
Twitter: @erikhatcher

# A word from your sponsor...

- Will Hayes talk:
  **Building and Scaling a High Performing Development Team**
  - with folks like you!: Passionate, Smart, Caring, Diverse
  - open source
  - Lucidworks IS HIRING!   More than doubled in the past year
- "[His proposed] fun, non-technical, talk [was] for members of development teams at all levels looking to increase the performance and productivity in themselves and others."

## you autocomplete us

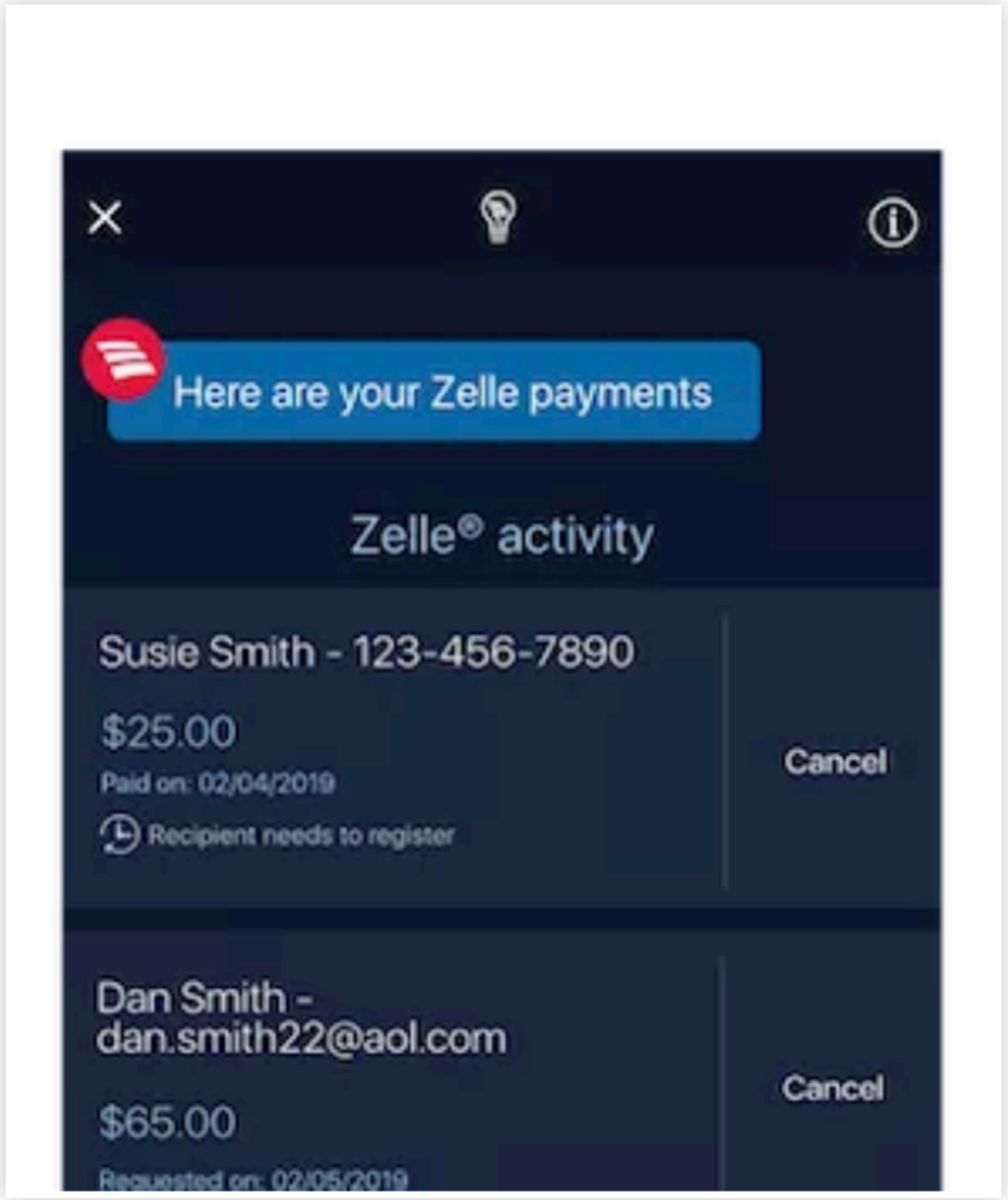
# Back to the talk...

This fun, technical, talk is for members of development teams at all levels looking to increase the relevancy and value in their search systems and data pipelines using Solr

# Bank of America's Intelligent Assistant Erica Helps Guide More Than 7 Million Customers

BY DEREK TOP *on* JUNE 10, 2019 · ( 0 )

One year after its nationwide launch, Bank of America's conversational intelligent assistant Erica has achieved significant milestones. Recently, BofA touted the success of its digital banking assistant that uses natural language processing and predictive analytics to engage digital banking customers. Among the notable achievements:


Here are your Zelle payments
Zelle® activity
Susie Smith - 123-456-7890
$25.00
Paid on: 02/04/2019
Recipient needs to register
Cancel
Dan Smith - dan.smith22@aol.com
$65.00
Requested on: 02/05/2019
Cancel

- Completed over 50 million client requests, including banking activities and more complex tasks
- 500,000 new users per month engage with Erica
- Expanded its knowledge base of financial questions from 200,000 at launch to more than 400,000 today
- Cross-generational engagement, with 15% from Gen Z, 49% millennials, 20% Gen X and 16% baby boomers/seniors

https://opusresearch.net/wordpress/2019/06/10/bank-of-americas-intelligent-assistant-erica-helps-guide-more-than-7-million-customers/

## Chatting with Solr

- Building a chat system relies on recognizing known entities being uttered. The Solr Tagger, a powerful unique capability of Apache Solr, provides performant tagging of known, concrete items in free text. Using the Solr Tagger as a first stage of query interpretation provides rich entity metadata that can be leveraged to hone in on user intent.

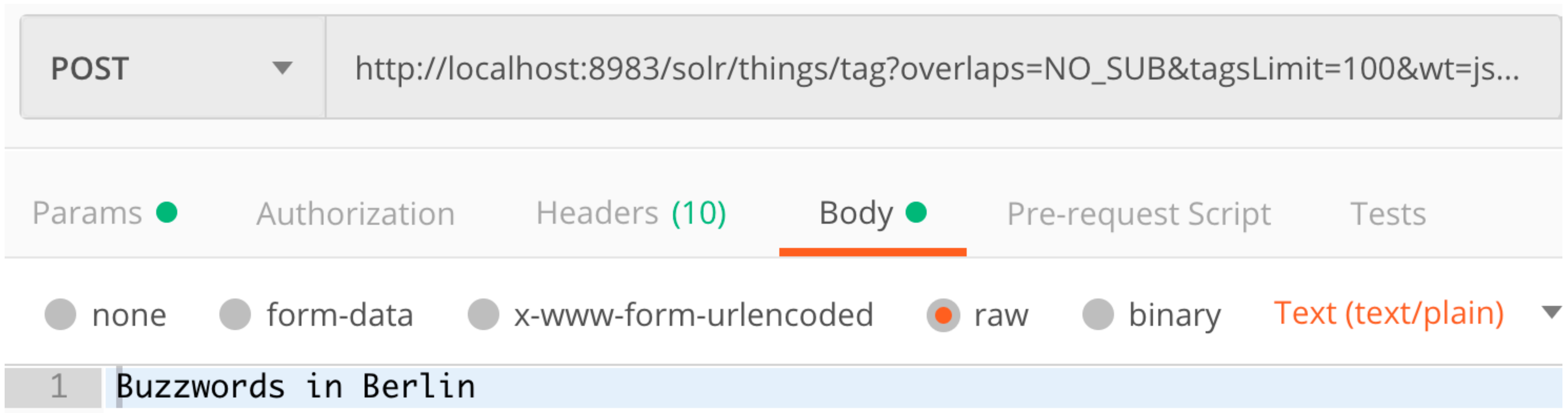
- This session will introduce the Solr Tagger and its myriad of use cases culminating with a chat application that creatively uses the tagger for query interpretation.

# Solr Tagger

- https://lucene.apache.org/solr/guide/the-tagger-handler.html
- Given a dictionary (a Solr index) with a name-like field, you can post text to this request handler and it will return every occurrence of one of those names with offsets and other document metadata desired. It's used for named entity recognition (NER).
- The tagger doesn't do any natural language processing (NLP) (outside of Lucene text analysis) so it's considered a "naive tagger", but it's definitely useful as-is and a more complete NER or ERD (entity recognition and disambiguation) system can be built with this as a key component. The SolrTextTagger might be used on queries for query-understanding or large documents as well.

# Tagger Basics

- straightforward tool to tag concrete, known (text string) entities
- "tags" are documents in a specialized collection
  - field type for tag fields must end with **`ConcatenateGraphFilterFactory`**
  - **`solr.TagRequestHandler`** end-point needs to be defined
- Index time, tagging document text
  - use tagged content for new fields
- Query time, tagging the query
  - use tags to modify the query for improved relevancy

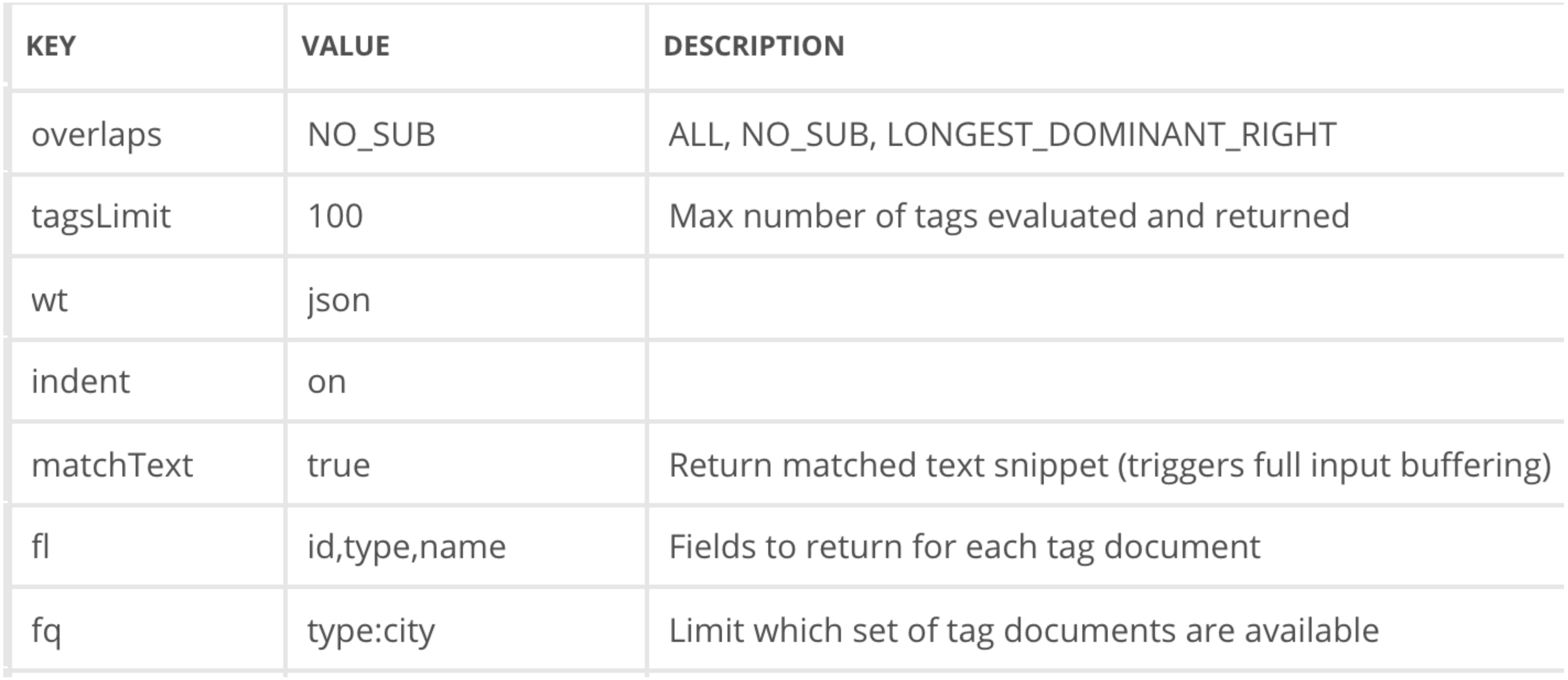


```
http://localhost:8983/solr/things/tag?overlaps=NO_SUB&tagsLimit=100&wt=js...
```

```
Buzzwords in Berlin
```

```
{
  "responseHeader":{
    "status":0,
    "QTime":0},
  "tagsCount":1,
  "tags":[[
      "startOffset",13,
      "endOffset",19,
      "matchText","Berlin",
      "ids",["2950159"]]],
  "response":{"numFound":1,"start":0,"docs":[
      {
        "type":"city",
        "name":["Berlin"],
        "id":"2950159"}]
  }}
```

# Tagger Parameters

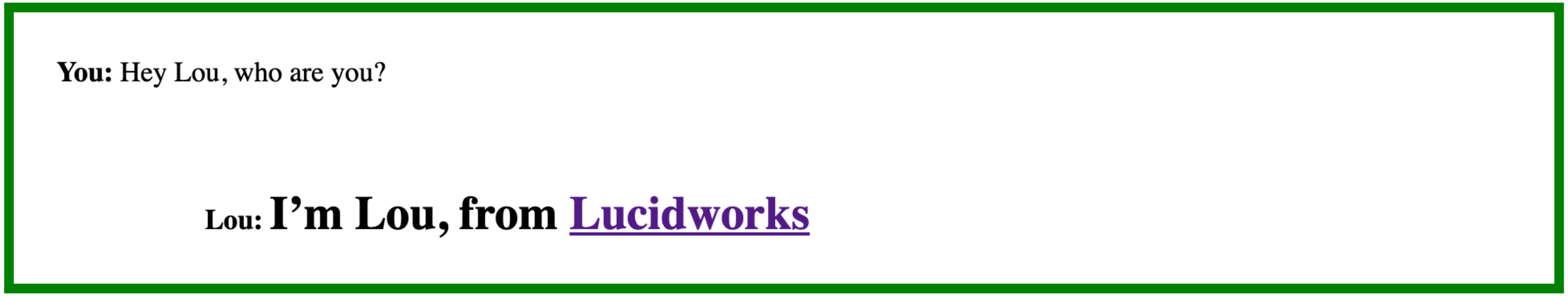


| KEY | VALUE | DESCRIPTION |
|---|---|---|
| overlaps | NO_SUB | ALL, NO_SUB, LONGEST_DOMINANT_RIGHT |
| tagsLimit | 100 | Max number of tags evaluated and returned |
| wt | json | |
| indent | on | |
| matchText | true | Return matched text snippet (triggers full input buffering) |
| fl | id,type,name | Fields to return for each tag document |
| fq | type:city | Limit which set of tag documents are available |

## Tagger in Perspective

- Use in combination with other techniques:
  - The tagger doesn't tag arbitrary dates or numbers; combine with a stage to recognize patterned content such as four digit numbers such as years: /\d\d\d\d/
  - Query context: location, user profile/demographics
  - NLP
- The tagger collection can be built from from static data (cities.csv) or machine learned (such as head/tail analysis)

# Meet Lou


**You:** Hey Lou, who are you?

**Lou:** **I'm Lou, from Lucidworks**

- Hey Lou, remind me soon to wrap up this demo
- Where is Berlin?
- What is the population of Berlin?

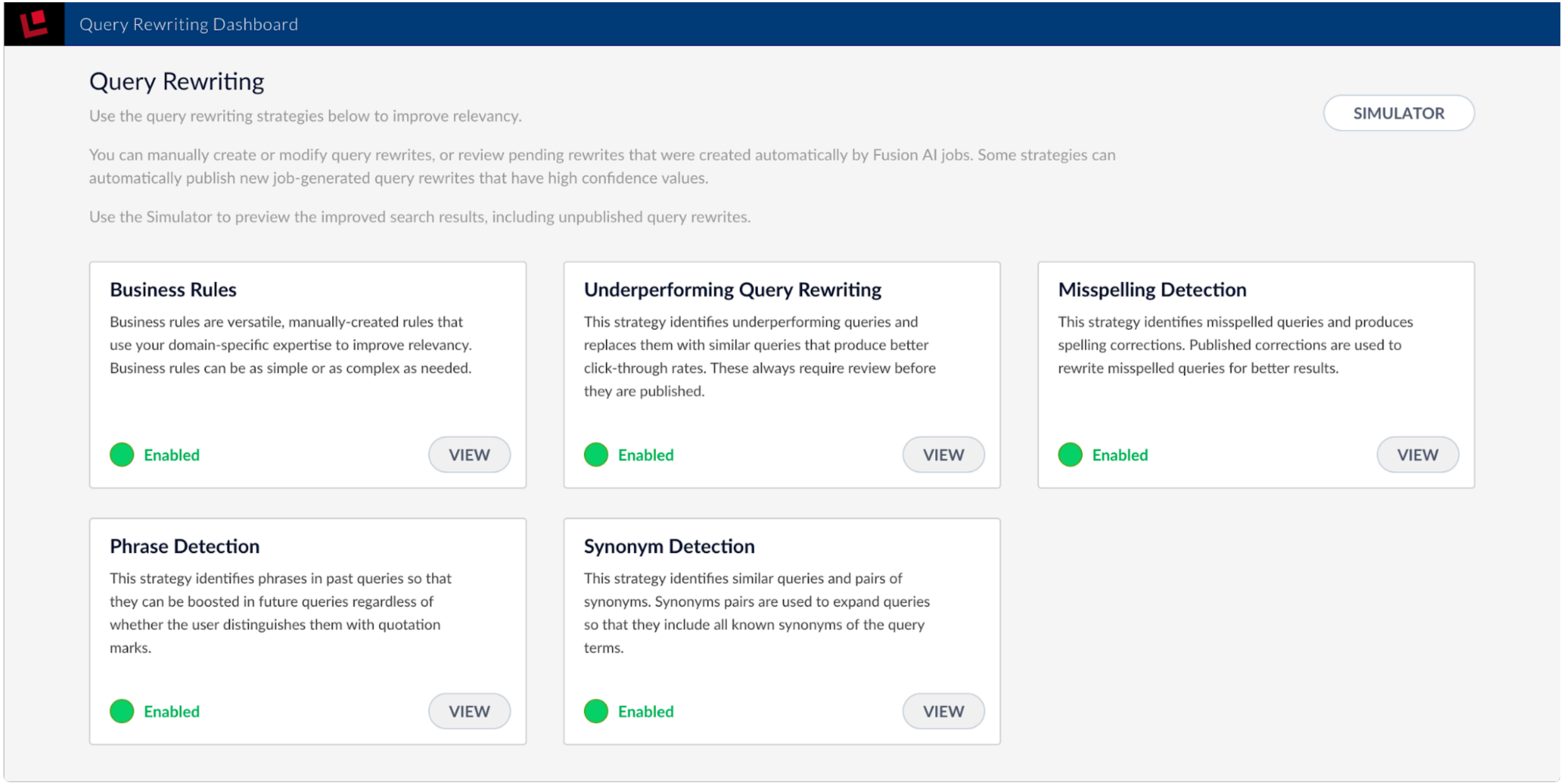
# Lou does...

- tagging/replacing of known **things** mentioned, by name, in utterances
    - basic information about a thing: id, type, name
    - things may contain any other information needed for findability or presentation
- looking up best **grammar** matches for tagged, typed utterances
- things: entities; typed documents in a Solr Tagger enabled collection
- grammar: slotted utterances and associated actions

# Lou's Query Pipeline

- Where is Berlin?
- Tagger: Where is Berlin?
  - {"name": "Berlin", "type": "city", "latitude_s": "52.52437", "longitude_s": "13.41053"}
- Grammar: Where is <city>?
  - render map to lat/long

# Machine Learned Tags in Fusion



Query Rewriting Dashboard

## Query Rewriting

Use the query rewriting strategies below to improve relevancy.

SIMULATOR

You can manually create or modify query rewrites, or review pending rewrites that were created automatically by Fusion AI jobs. Some strategies can automatically publish new job-generated query rewrites that have high confidence values.

Use the Simulator to preview the improved search results, including unpublished query rewrites.

### Business Rules

Business rules are versatile, manually-created rules that use your domain-specific expertise to improve relevancy. Business rules can be as simple or as complex as needed.

● Enabled — VIEW

### Underperforming Query Rewriting

This strategy identifies underperforming queries and replaces them with similar queries that produce better click-through rates. These always require review before they are published.

● Enabled — VIEW

### Misspelling Detection

This strategy identifies misspelled queries and produces spelling corrections. Published corrections are used to rewrite misspelled queries for better results.

● Enabled — VIEW

### Phrase Detection

This strategy identifies phrases in past queries so that they can be boosted in future queries regardless of whether the user distinguishes them with quotation marks.

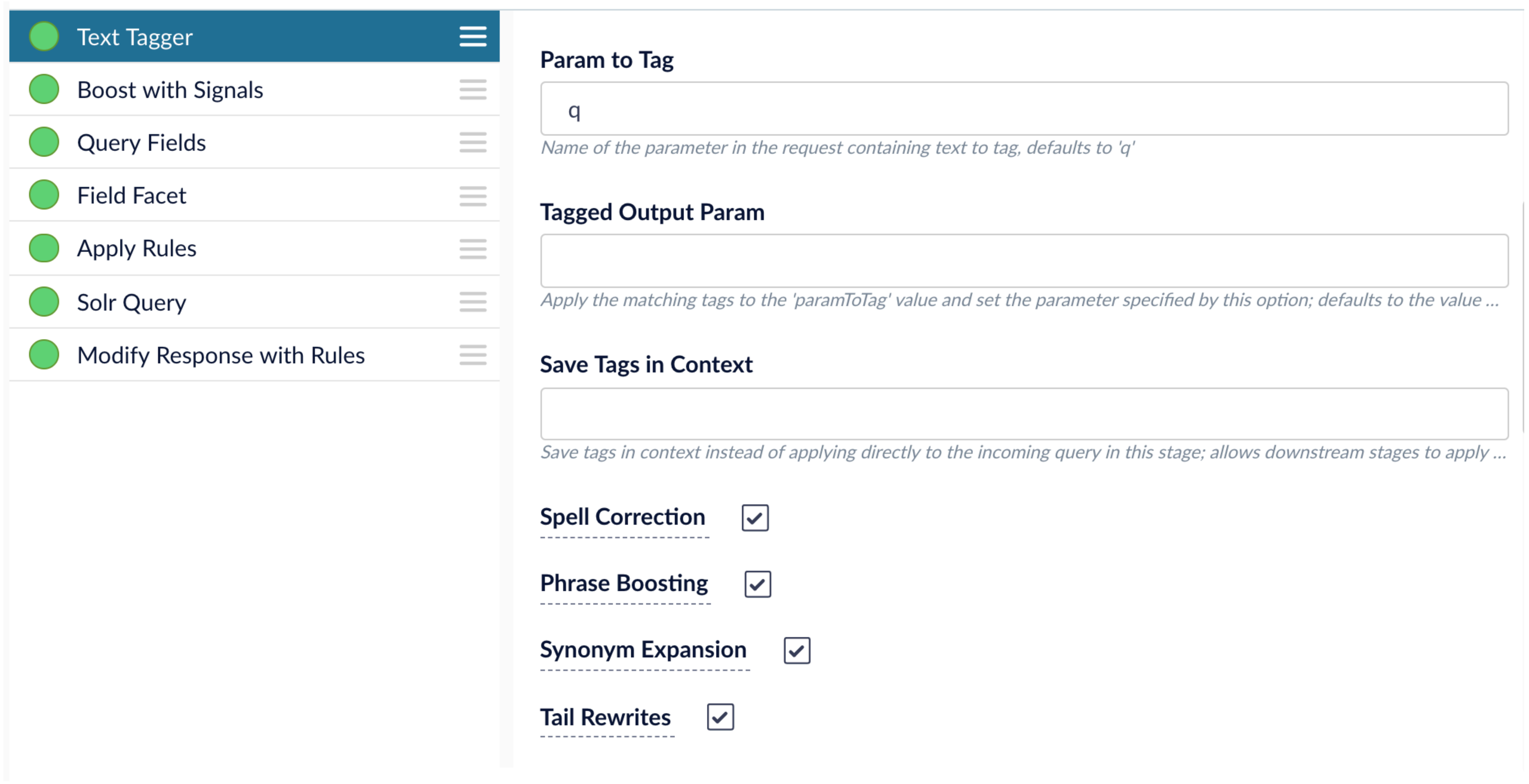
● Enabled — VIEW

### Synonym Detection

This strategy identifies similar queries and pairs of synonyms. Synonyms pairs are used to expand queries so that they include all known synonyms of the query terms.

● Enabled — VIEW

# Tagging in Lucidworks Fusion



**Param to Tag**

q

*Name of the parameter in the request containing text to tag, defaults to 'q'*

**Tagged Output Param**

*Apply the matching tags to the 'paramToTag' value and set the parameter specified by this option; defaults to the value ...*

**Save Tags in Context**

*Save tags in context instead of applying directly to the incoming query in this stage; allows downstream stages to apply ...*

**Spell Correction** ✓

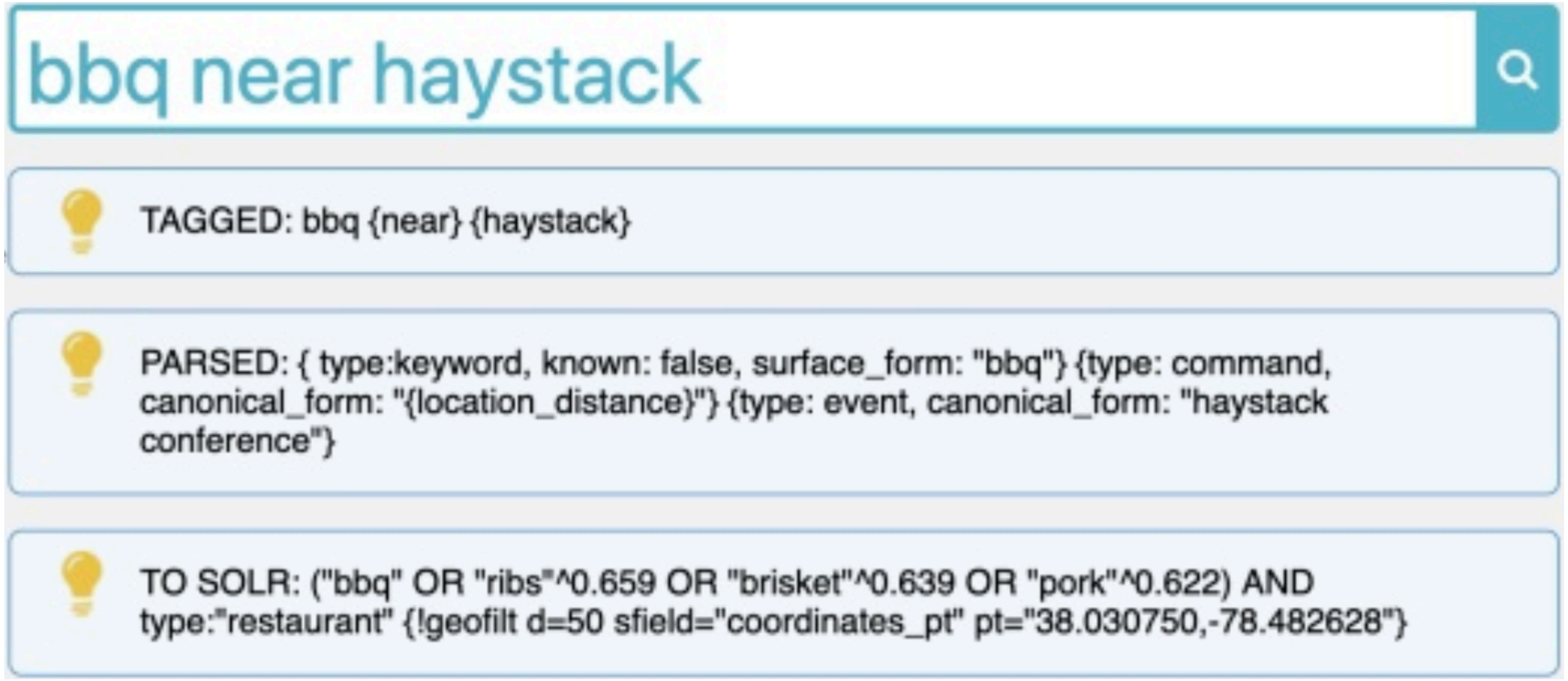
**Phrase Boosting** ✓

**Synonym Expansion** ✓

**Tail Rewrites** ✓

## See also...

- Using Solr Tagger for Improving Relevancy
    - https://lucidworks.com/2019/05/17/solr-tagger-improving-relevancy/


bbq near haystack

TAGGED: bbq {near} {haystack}

PARSED: { type:keyword, known: false, surface_form: "bbq"} {type: command, canonical_form: "{location_distance}"} {type: event, canonical_form: "haystack conference"}

TO SOLR: ("bbq" OR "ribs"^0.659 OR "brisket"^0.639 OR "pork"^0.622) AND type:"restaurant" {!geofilt d=50 sfield="coordinates_pt" pt="38.030750,-78.482628"}

## Natural Language Search with Knowledge Graphs presented by Trey Grainger

https://haystackconf.com/2019/natural/

## Conclusion

- Solr Tagger provides known entity tagging of text
- Very fast, even on large text
- A great first stage in a query pipeline, providing valuable metadata to then use cleverly to enhance query interpretation and result relevancy
- Useful as an index pipeline pre-processor to extract known entities from text, pulling them into additional fields for faceting.  Naive "classifier"?