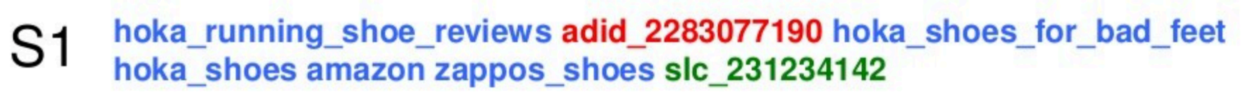
Search2Vec

Main Idea

Using search sessions as a set of related actions.

Actions ——> (Queries, Add clicks, Link clicks)

Example:



S1 hoka_running_shoe_reviews adid_2283077190 hoka_shoes_for_bad_feet hoka_shoes amazon zappos_shoes slc_231234142

Then, deriving embeddings (between queries and ads). Training like Skip-Gram model with negative sampling including

1. dwell time (+ & - impact)
2. skipped ads (- impact)

Vocabulary

All frequent search queries appeared in the training set + All Ads (ad ids) + Links (Not Sure)

Goal

Given a query, finding its nearest ad in order to get as much ad clicks as possible.

Sub-Goals

- **Memory Problem**
    - Distributed systems (Not crucial)

- **Cold Ads (Not yet seen ads)**

  - Extracting important phrases and words (all n-grams) out of ad context (bid term, title, description and ⋯), looking for identical vectors for them in query vocabulary, summing (the similar ones to bid term vector) it all up to find a new vector.

```
Data: ad title, ad description, ad URL, bid term $b$,
      threshold $\tau_c$, set of context query vectors $V_q$
Initialization: $\tau_c = 0.45$, get bid term vector $\mathbf{v}_b$ from
$V_q$, set $\mathbf{v}_{ad} = \mathbf{v}_b$, create ad document $\mathbf{d}_{ad}$ from all
$n$-grams $(n = 1, .., 10)$ in ad title, description, URL;
forall the phrases $p$ in ad document $\mathbf{d}_{ad}$ do
    if $\mathbf{v}_p$ exists in $V_q$ then
        if $sim(\mathbf{v}_b, \mathbf{v}_p) > \tau$ then
            $\mathbf{v}_{ad} = \mathbf{v}_{ad} + \mathbf{v}_p$;
        end
    end
end
Result: ad content vector $\mathbf{v}_{ad}$
```

Algorithm 1: content search2vec

  - How to find those effective n-grams? using Algorithm 1; anchor phrase model. For more than 80 percent of new ad's bid terms, the corresponding vector for the bid term is available. If it's not, we can use Tale Query method for the ad to allocate a vector for its content.

    —> content serach2vec

- **Tale Queries (Not seen in the query space)**

  - Constructing a dataset of head queries related to its 10 most relevant queries. then searching for the input query words in this dataset. The most similarity will correspond to the proper head query. Then the obtained query's ad suggestion, will be applied to the new query.
  - For a new query: textual match (?) against document, retrieve vector from top match —> elastic serach2vec

What to do with “Torob” datasets?

1. Search logs: convert it into related sessions, using date and time, userId and ?
2. Click logs: external key to searchId, external key to productId, date and rank.
3. Products: trivial

We can construct sessions using Search query associated with its Clicks. Also, we can add different queries which their only difference is in “page” (dates should be close).