Long Tail Query Enrichment for Semantic Job Search

By: Layla Pournajaf (Facebook)
Khalifeh AlJadda (CareerBuilder)
Mohammed Korayem (CareerBuilder)

Search Data Science @CareerBuilder

CAREER BUILDER™

About Me

Khalifeh AlJadda

Lead Data Scientist, Search Data Science

CAREERBUILDER™

- Joined CareerBuilder in 2013 as intern
- **PhD, Computer Science – University of Georgia**
- **BSc, MSc, Computer Science, Jordan University of Science and Technology**

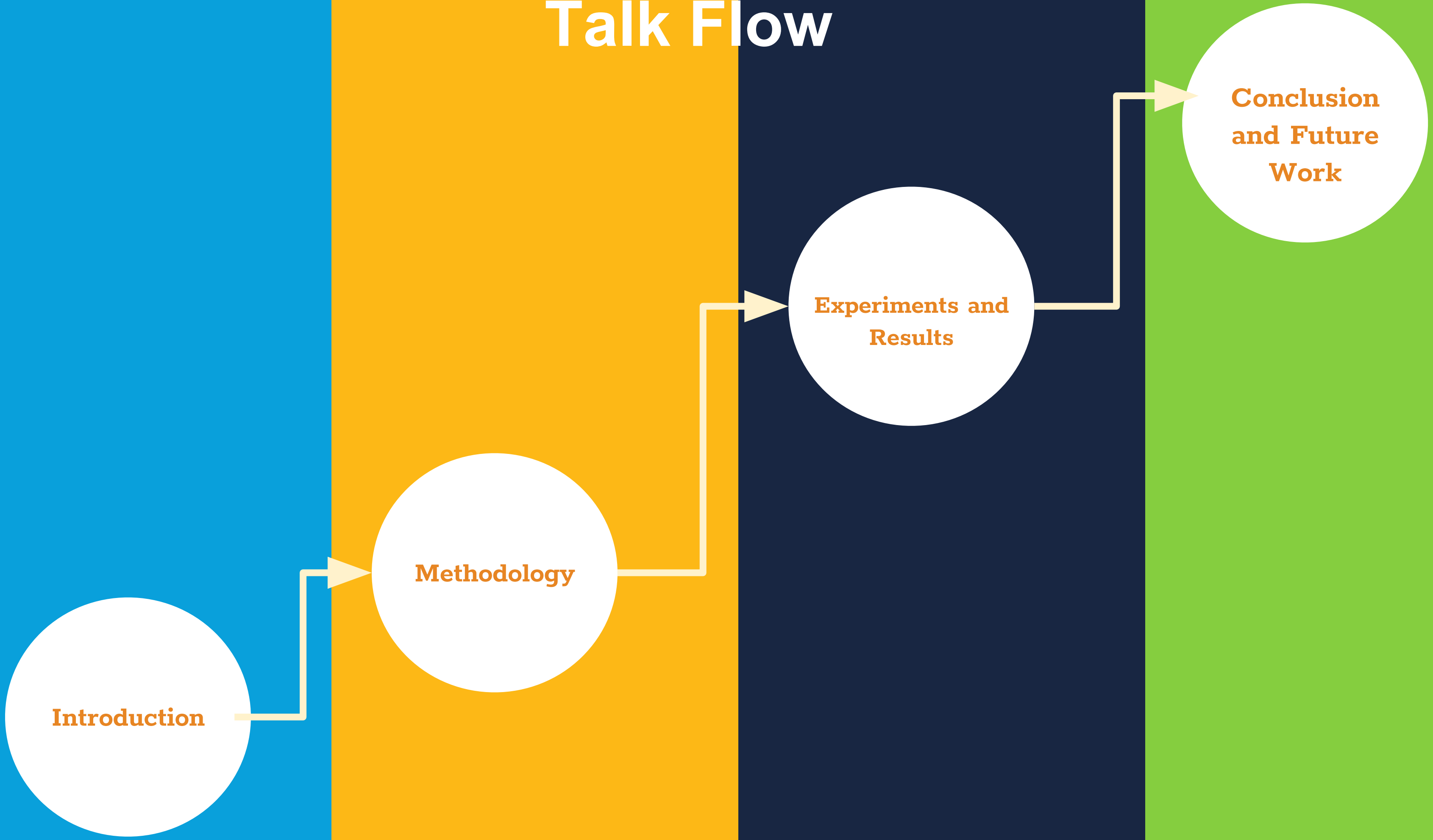
Activities:

- Founder and Chairman of Southern Data Science Conference (www.southerndatascience.com)
- Frequent public speaker in data science and big data analytics domains
- Creator of GELATO (Glycomic Elucidation and Annotation Tool)

Check the CFP of our first Research Track

Southern Data Science Conference

Talk Flow


Introduction
Methodology
Experiments and Results
Conclusion and Future Work

Keyword-based Search

- Traditional search engines (i.e. Lucene, Solr, Elasticsearch) tokenize text and find documents containing those tokens and linguistic variations:
    - User's Search: **machine learning**
      Tokenization: ["**machine**", "**learning**"] =>
      Stemming: ["machin", "learn"]
      Final Query: **machin AND learn**
      This could match a document for a "machinist" who has "learned" something.
    - **software architect** => … => **software AND architect**
      Might identify a **building architect** requiring knowledge of specialized architecture **software**

Semantic Search (Search for Things not Strings)

- We need a way to identify and search for the **meaning of keyword phrases**, not just the individual text tokens
  - i.e. machine learning = "machine learning" OR "data scientist" OR "mahout" OR "svm" OR "neural networks"

Our Target

User's Query:

machine learning research and development Portland, OR software engineer AND hadoop, java

Traditional Query Parsing:

(machine AND learning AND research AND development AND portland) OR (software AND engineer AND hadoop AND java)

Semantic Query Parsing:

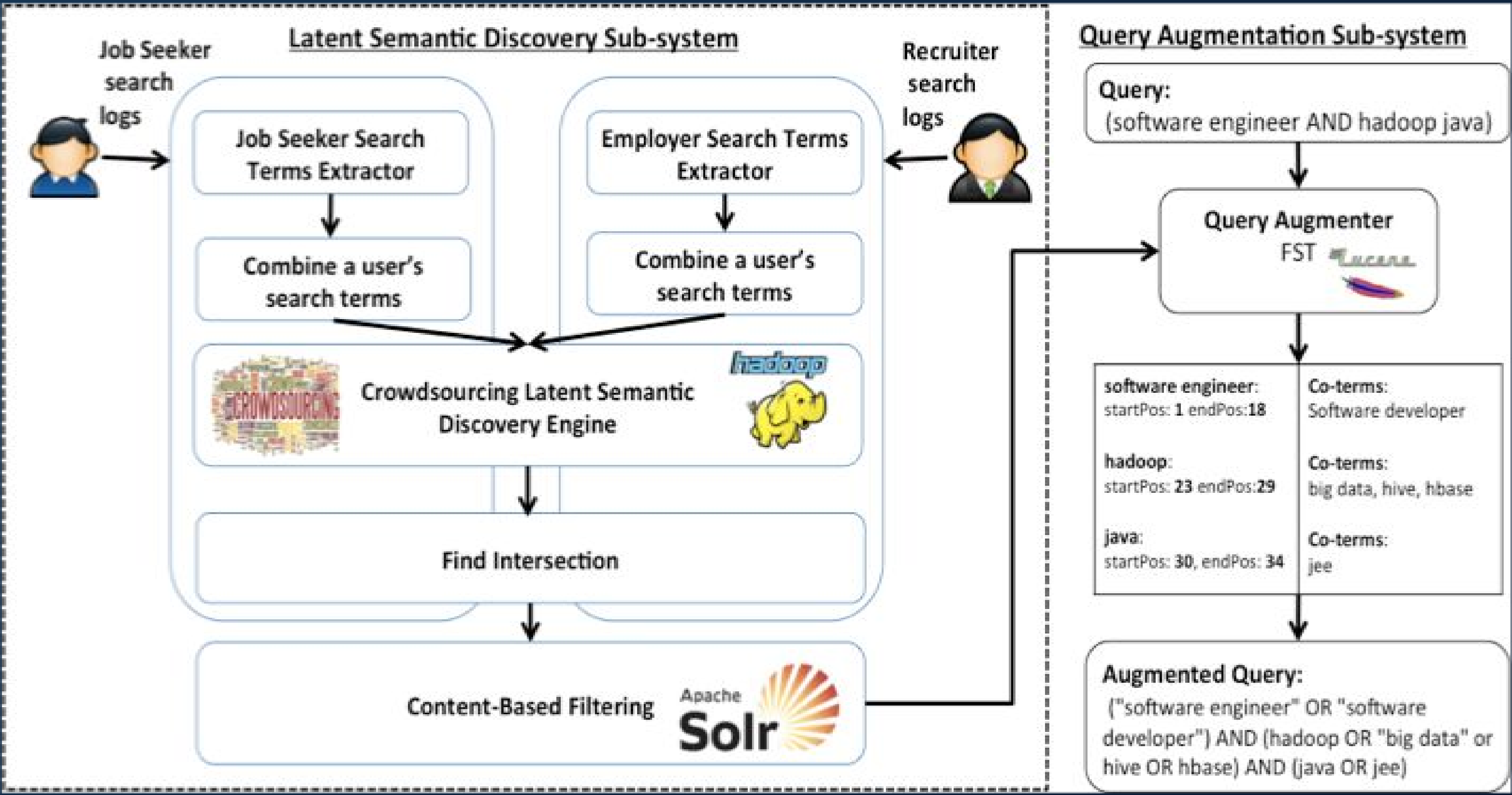
"machine learning" AND "research and development" AND "Portland, OR" AND "software engineer" AND hadoop AND java

Our Target

<u>Semantically Expanded Query:</u>

("machine learning"^10 OR "data scientist" OR "data mining" OR "artificial intelligence")
AND ("research and development"^10 OR "r&d") AND
AND ("Portland, OR"^10 OR "Portland, Oregon" OR {!geofilt pt=45.512,-122.676 d=50 sfield=geo})
AND ("software engineer"^10 OR "software developer")
AND (hadoop^10 OR "big data" OR hbase OR hive) AND (java^10 OR j2ee)

+

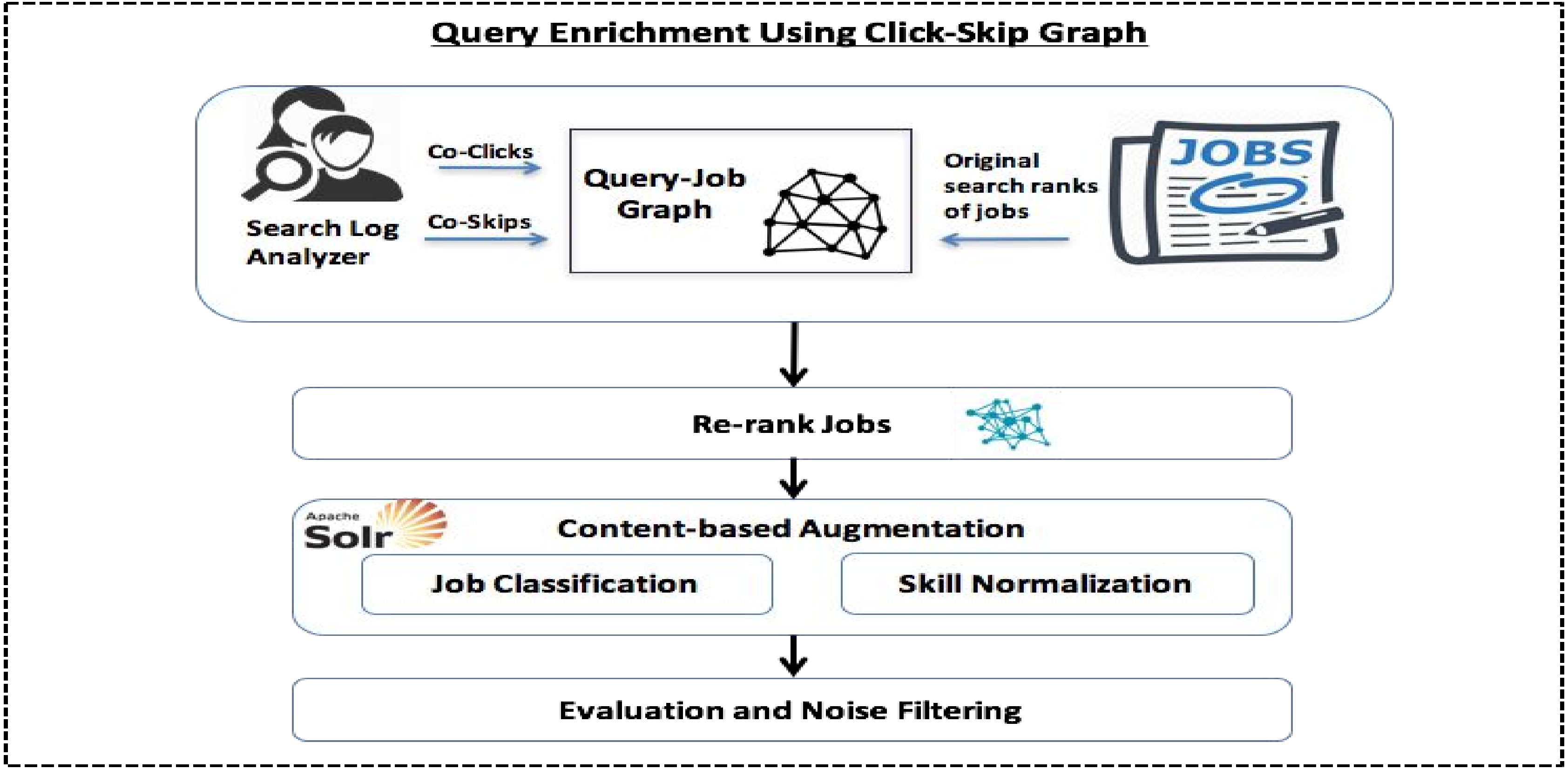


*Crowdsourced Query Augmentation through Semantic Discovery of Domain-specific Jargon

Problem Description

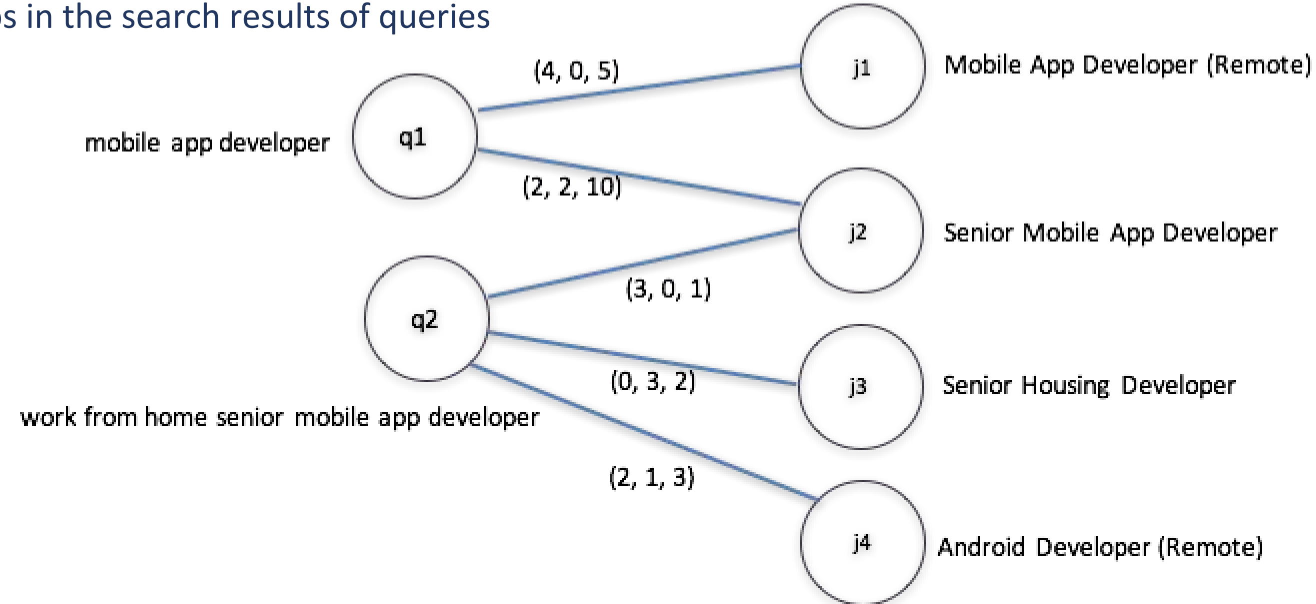
- The typical distribution of query frequencies is very heavy on the tail which results in a large set of unpopular or uncommon queries.
- Although, tail queries are individually uncommon, they make up a large portion of queries collectively.
- Ignoring such heavy long tails in the job search domain may result in turning away job seekers who are not able to find relevant jobs.
- Our approach of using the wisdom-of-the-crowd via the co-occurrence of queries can't enrich such long tail queries.

Methodology



Click-Skip Bipartite Graph

- Nodes of our click-skip graph comprise of jobs and queries while edges capture explicit and implicit behavior of query issuers.
- We log the co-clicks –when a job is clicked as a result of a query, and co-skips –when a job appears in the result set of a query but is skipped for a job with higher ranking. We also record the original rankings of jobs in the search results of queries



Re-Ranking the matched jobs

- We re-rank the retrieved jobs for each query using the collected signals on the edges of the click-skip graph.
- we assume the combined click-skip score follows a beta distribution.
- we expect that the jobs with higher rankings receive more clicks than skips, so we can use ranking values to construct our prior belief parameters $a = c_r$ and $b = s_r$ for every ranking.
-

$$c_r = \left(\frac{1-\mu_r}{\sigma_r^2} - \frac{1}{\mu_r}\right)\mu_r^2$$

$$s_r = c_r\left(\frac{1}{\mu_r} - 1\right)$$

Cont..

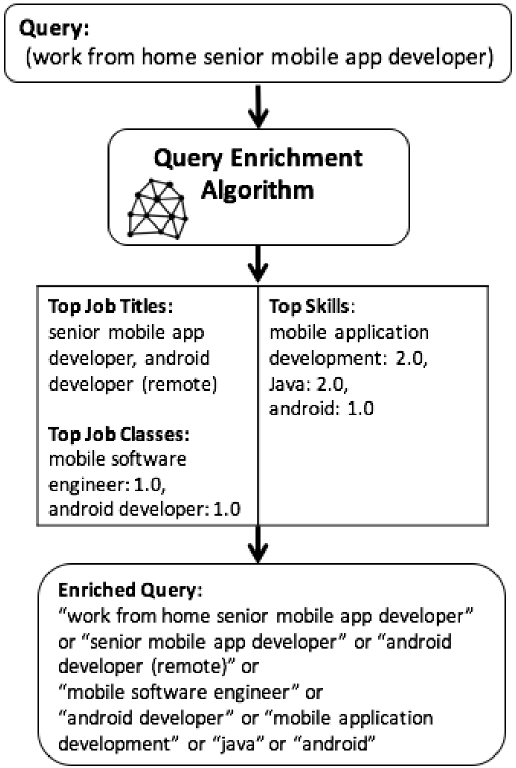
- Next, we estimate the posterior distribution of our aggregate score by applying the observed co-Clicks and co-Skips .

- Finally, we compute the expected aggregate score for every query job pair (q, j) with ranking of r = $ranking_{qj}$ as below:

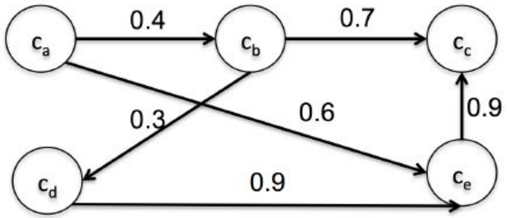
$$aggScore_{qj}^{(r)} = E(\beta(c_r + coClicks_{qj}, s_r + coSkips_{qj})) = \frac{c_r + coClicks_{qj}}{c_r + coClicks_{qj} + s_r + coSkips_{qj}}$$

Content-based Augmentation

Query Enrichment Example



- We select the top 3 jobs after re-ranking.
- For each job we send it to our inhouse NLP parser.
- The parsed jobs then sent to the in-house job title classification and skills normalization.
- The normalized job title(s) and skills have to pass a validation phase before they are used to enrich the related long-tail query.



Experiment and Results

TABLE I: Click-Skip Graph Summary

# of Queries	# of Popular Queries	# of Tail Queries	# of Jobs	# of Edges
25.8K	12.4K	13.4K	200K	400K

TABLE II: Enrichment Relevancy Evaluation

Method	Coverage [w/o filtering]	Coverage [w/filtering]	Avg. Relevancy Score
No-prior	98.04%	88.53%	0.79
Rank-based prior	98.04%	88.76%	0.83

Conclusion and Future Work

- we propose a method for semantic augmentation of long tail queries in the job search domain using clickthrough and search logs.

- Our method identifies top relevant jobs by analyzing the clicks and skips signals of job seekers and extracts their classifications to generate richer queries.

- we plan to extend this method by building a query embedding with a more focus on the contents of both the tail queries and clicked jobs.

We Are Hiring
(PhD Interns for Spring)

URL: www.aljadda.com
Twitter: @aljadda
Email: khalifeh.aljadda@careerbuilder.com

CAREER BUILDER™