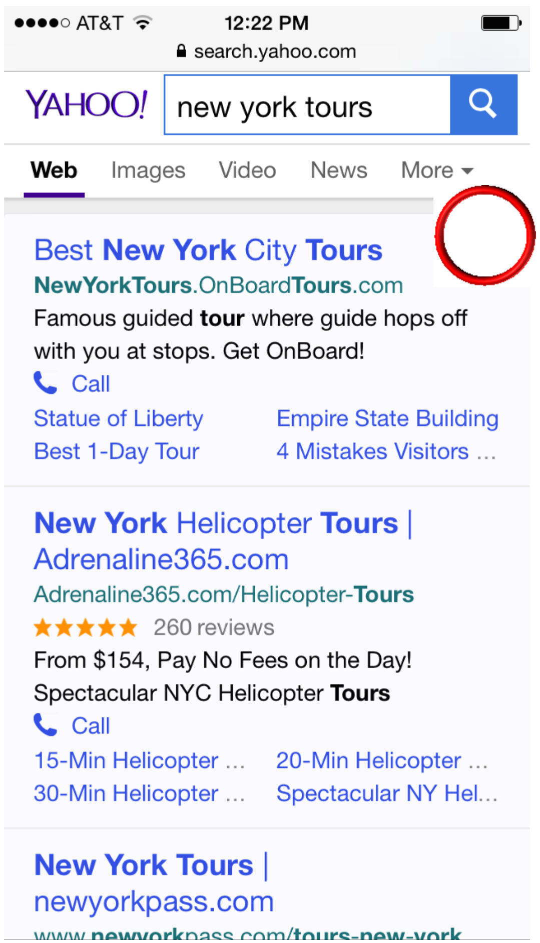
Scalable Semantic Matching of Queries to Ads in Sponsored Search

Mihajlo Grbovic, Nemanja Djuric, Vladan Radosavljevic, Fabrizio Silvestri, Ricardo Baeza-Yates, Andrew Feng, Erik Ordentlich, Lee Yang, Gavin Owens

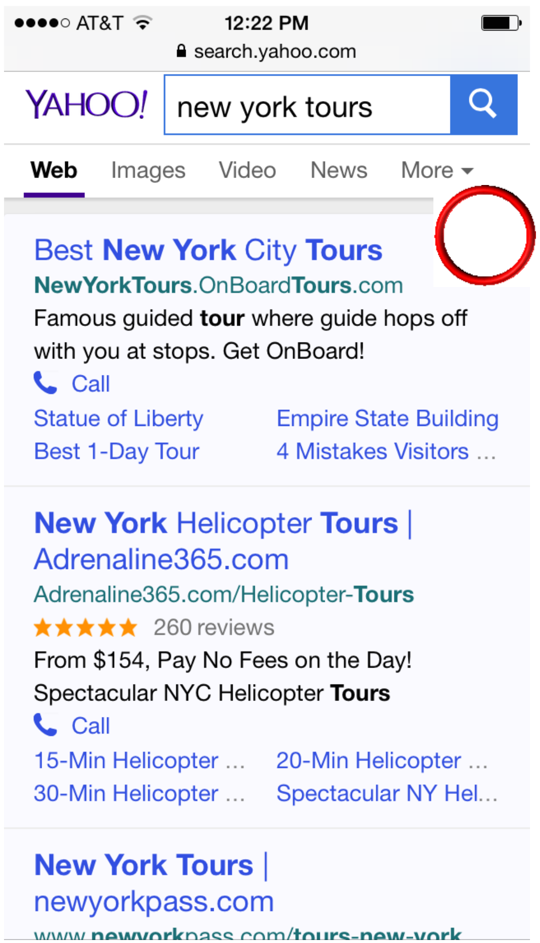
Yahoo Research, Advertising Sciences

Sponsored Search



- **Advertisers**
  - Provide ad creative (title, description, url)
  - Provide bidterms (queries they want their ad to show for)
- **Search Engine**
  - Matches queries to bidterms (exact match + variant match)
  - Implements: **broad match**

Sponsored Search



Broad Match: advanced matching to non-provided keywords by:

- **Query rewriting:**
  - Given a user query, find K semantically similar queries
- **Query-ad matching:**
  - Need to place queries and ads in same feature space

Query and Ad Representations

How to represent queries and ads?

1) Traditional – Bag of words

1 – where query words are
0 – everywhere else

Query:
cheap flights

bag of words vector:

0	1	0	1	0	…	0	0
	cheap		flights				

Ad:
Get the best air ticket deals

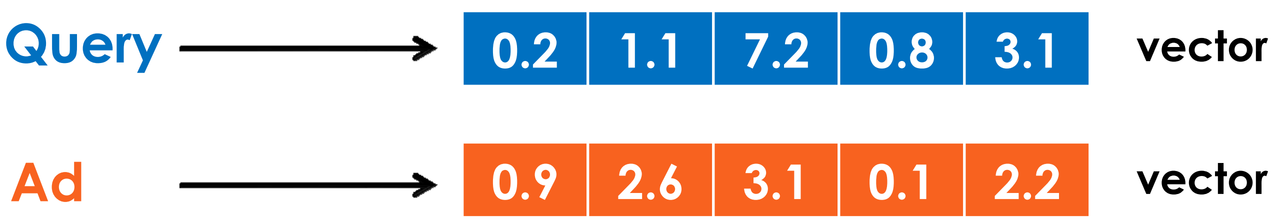
bag of words vector:

1	0	1	0	1	…	1	1
air		ticket		get		deals	best

ISSUE – No way we can find that this query and this ad are related

Query and Ad Representations

2) New – move from sparse to dense vectors



- Represent **queries** and **ads** as **numeric vectors**
- Vectors need to be learned using training data (**search sessions**)
- We want queries/ads with **similar contexts** to have **similar vectors**

session 1:	trip_ideas	**cheap_flights**	holiday_travel_deals
session 2:	trip_ideas	**air_tickets**	holiday_travel_deals

Search2Vec

search2vec = word2vec [1] where:

- **words = {queries, search ads, search links}**
- **documents = search sessions** (uninterrupted sequences of user actions on the search engine)

[1] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of NIPS, 2013

Search2Vec

Search Sessions Dataset

S1 hoka_running_shoe_reviews adid_2283077190 hoka_shoes_for_bad_feet hoka_shoes amazon zappos_shoes slc_231234142

S2 king_tut king_tut_exibit king_tut_exibit_seattle adid_3858375378

S3 gas_caps gas_cap_replacement_for_cars adid_1066604760 gas_door_replacement_for_cars slc_81285142 fuel_door_covers autozone_auto_parts adid_253157233

S4 hoka_one_one run_florida hoka_shoes shoes_with_sl_2_last shoes_with_a_bigger_toe_box stans_shoes clarks_shoes slc_1567342

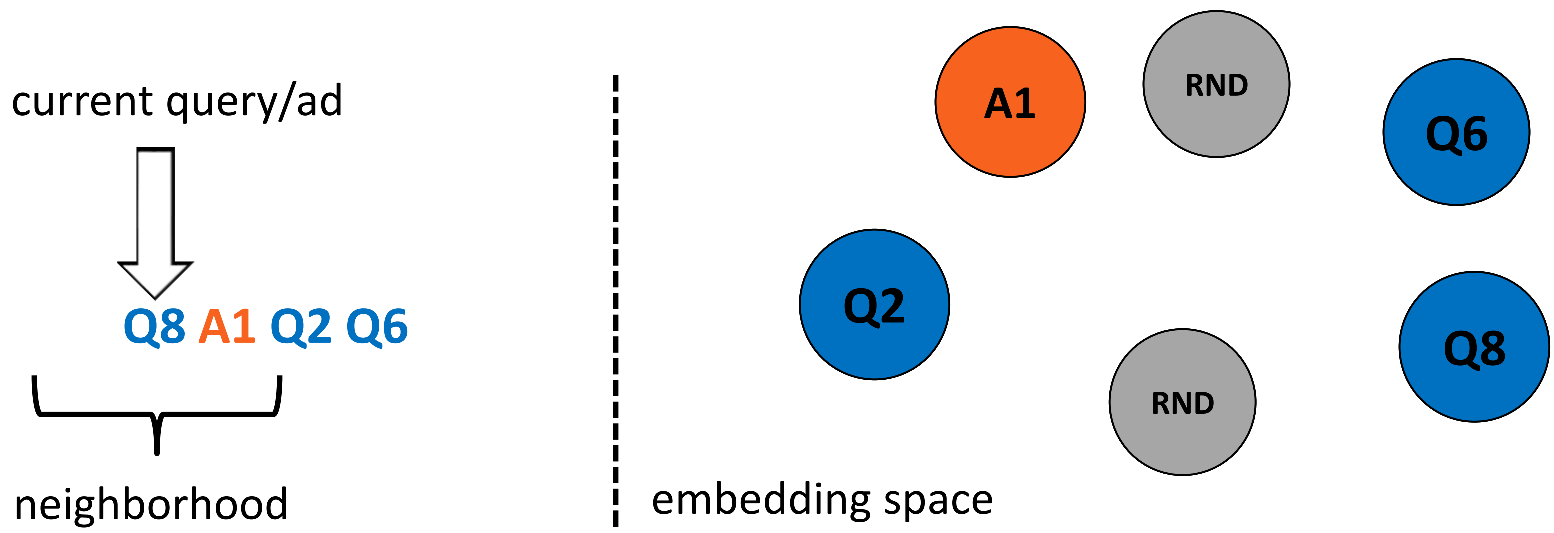
Search2Vec

Example search session:

Query8, **Ad1**, **Query2**, **Query6**

$$\mathbf{v}_i^{new} = \mathbf{v}_i + \eta \cdot (1 - \sigma(\mathbf{v}_i^T \mathbf{u}_{neigh})) \cdot \mathbf{u}_{neigh}$$

$$\mathbf{v}_i^{new} = \mathbf{v}_i - \eta \cdot \sigma(\mathbf{v}_i^T \mathbf{u}_{neg}) \cdot \mathbf{u}_{neg}$$



Search2Vec – after training

Query-to-Query similarity

cheap_flights

0.2	1.1	7.2	0.8	3.1

cosine similarity=0.9

air_ticket_deals

0.21	1.2	6.8	0.74	3.2

Search2Vec – after training

Query Rewriting

Try our state-of-the-art system for query rewriting.

scuba

Result	Relevance
No results	

Input Type:

Search Query

Output Type:

Search Query

Search

Search2Vec – after training

Query-to-Ad matching

ad_243609_341454

0.2	1.1	7.2	0.8	3.1

similarity=0.871

0.2	1.2	6.8	0.7	3.2

mystery_games

ad metadata	**ad id**: 243609 **bidterm id**: 341454 **ad title**: Host a Fun Murder Mystery Party **ad description**: Huge selection of fun murder mystery games for all ages, groups. #1 site for instant downloads and boxed sets of exciting murder mystery party games
generated keywords	murder mystery murder mystery games mystery games murder mystery party mystery party free murder mystery games for parties detective games murder mystery game how to host a mystery party for kids murder mystery dinner friends game night murder mystery parties at home murder mystery dinner party ...

Search2Vec – additional context

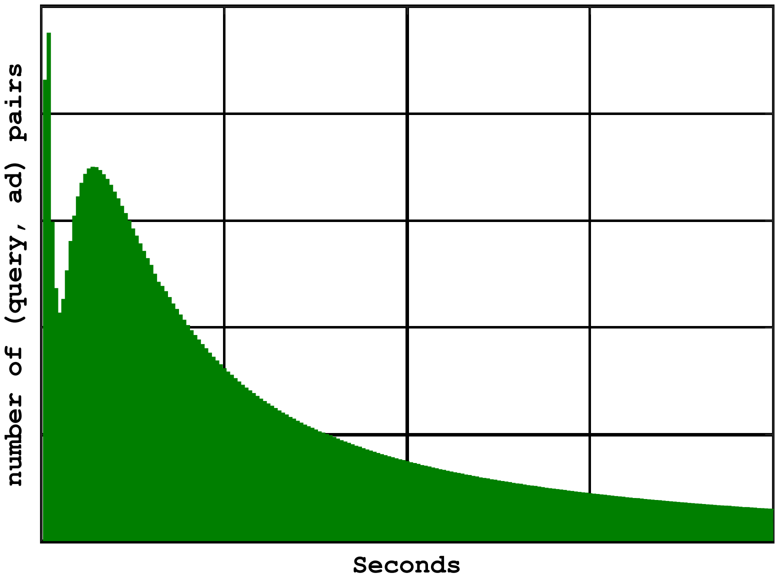
- How can we leverage additional **search-specific context**?
- What is context for a query in web search?

1. **Other queries** – in user search sessions
2. **Ads** - clicked (positives), **dwell time**, **skipped ads (negatives)**
3. **Search results** - clicked (represented with url)

Search2Vec – additional context

Dwell-time sensitive updates

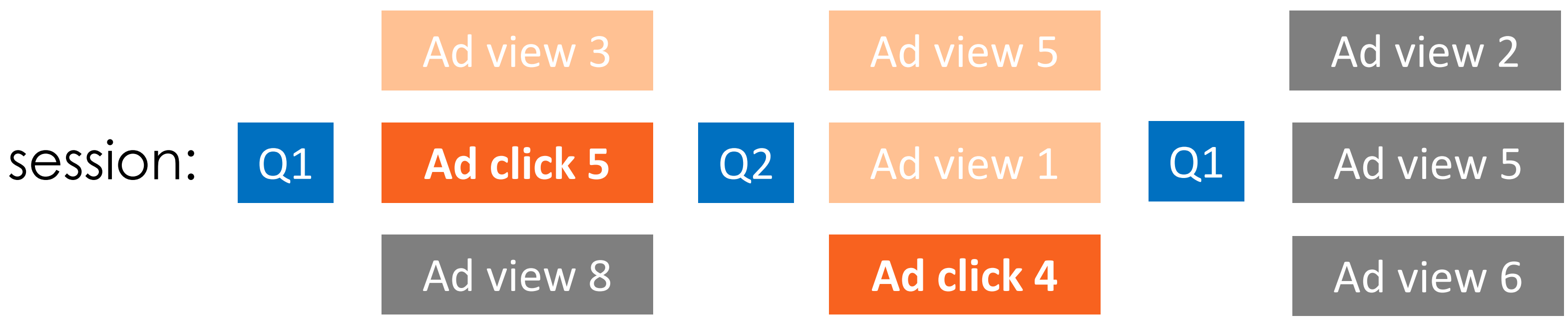
- gradient multiplier: **ni = log(1+t)**
  t=dwell time in minutes
- ad clicks with longer dwell time
  -> **larger learning rate**
- ad clicks with short dwell time
  -> **small learning rate**



Search2Vec – additional context

Ad skips as implicit negative signal

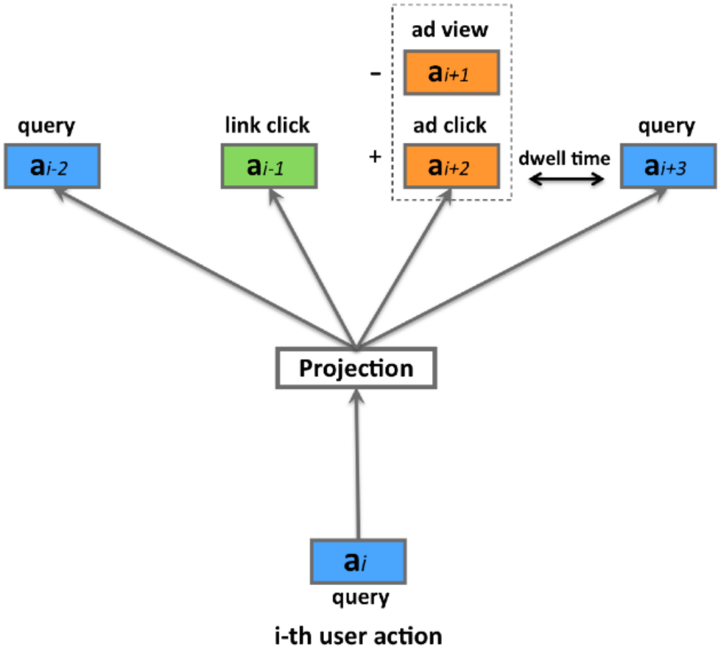
- Skipped ads = ads at high positions skipped in favor of ad click at lower position



Search2Vec – final model

- **a** = action (**q**, **ad**, **slc**)
- **D** = immediate context as positives
- **Dr** = random negatives (5 per session)
- **Dn** = implicit negatives (skipped ads)

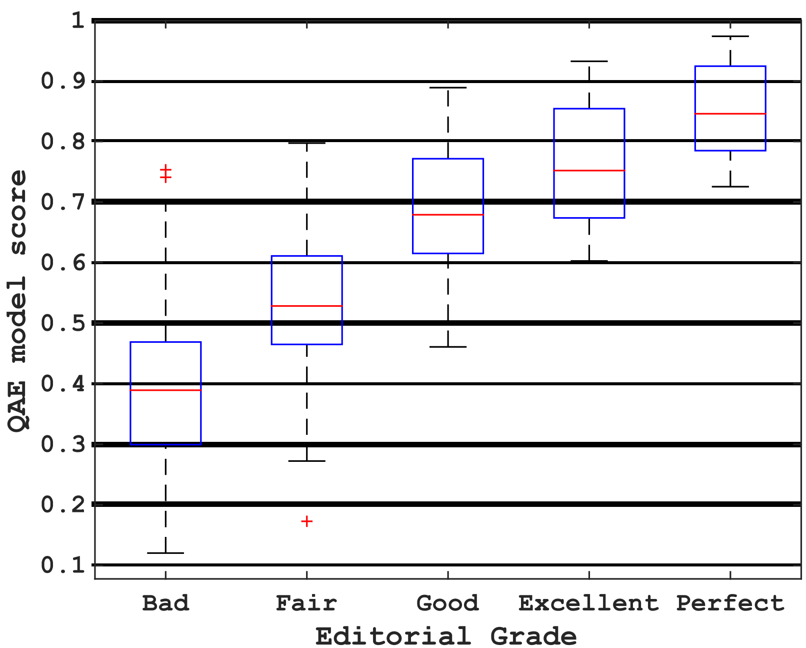
- **v = 300 dim vector**
- **c = 5 (context window size)**
- **10B sessions -> 80M vectors**

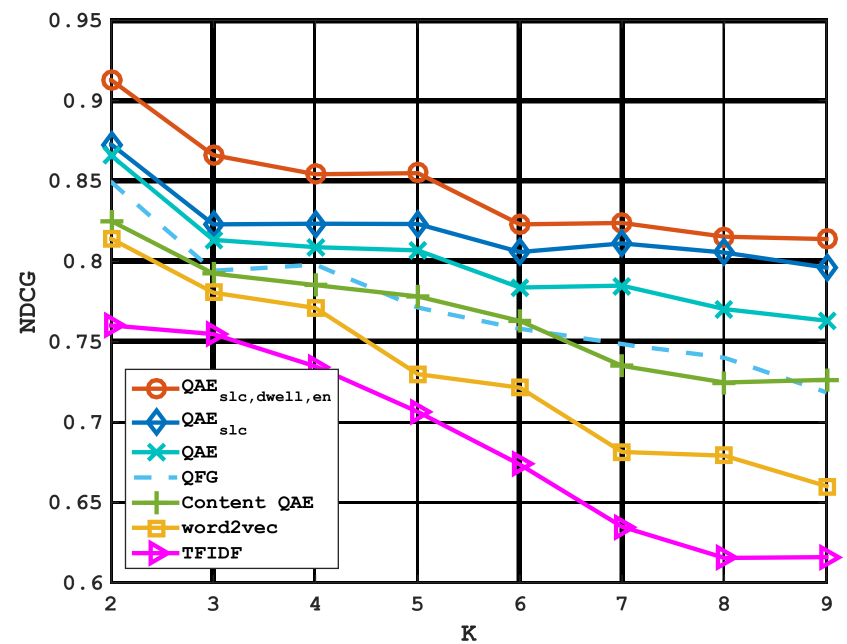


$$\underset{\theta}{\operatorname{argmax}} \sum_{(a,c)\in D} \log \frac{1}{1+e^{-v_c \cdot v_a}} + \sum_{(a,c)\in Dr} \log \frac{1}{1+e^{v_c \cdot v_a}} + \sum_{(q,ad)\in Dn} \log \frac{1}{1+e^{v_q \cdot v_{ad}}}$$

Search2Vec - editorial evaluation

- 20K judgments <**query**, **ad**, **score**, **grade**> :
  - **grade** = {Bad, Fair, Good, Excellent, Perfect}
  - <**cheep tickets**, **travelocity ad**, **0.831**, **Perfect**>





Search2Vec – A/B test

- **2 ways to increase Revenue Per Search:**

1) Increase Depth: find more ads for queries that have ads

Query 1	Ad 1	Ad 2	Ad 3	Ad 4	Ad 5
Query 2	Ad 6	Ad 7			

2) Increase Coverage: find ads for queries that do not have ads

Query 3	Ad 8	Ad 9
Query 4	Ad 5	

Ads	existing ads
Ads	**new ads**

Search2Vec – A/B test

- For each **query** find closest 30 **ads** in embedding space **above 0.7 similarity** and store in a **<query, ad list> table**
- **Control**: does not include this table: **Ads**
- **Bucket**: includes this table: **Ads** + **Ads**

Bucket	Query Coverage	Auction Depth	CTR	Click Yield	Revenue per Search
1-machine	**+1.14%**	**+2.13%**	**+0.5%**	**+1.7%**	**+7.07%**

- Low overlap with other match types: **90% pairs are unique**

Search2Vec – Limitations

1. **size of vocabulary**
   - **problem**: single 256GB machine can train up to 80M vectors
   - **solution**: distributed training
2. **cold ads**
   - **problem**: new ads added daily (no clicks to train ad vectors)
   - **solution**: content vectors (create ad vectors from ad text)
3. **tail queries**
   - **problem**: not enough observations to train a query vector
   - **solution**: index head query vector-based expansions

Distributed Search2Vec

- **Training**
  - Initialize pair of vectors **v** (input) and **u** (output) for each word in vocab
  - Update **v** of **center word** and **u**'s of **neighbors** and **random negatives**

w1 w2 w3 w4 **w5** w6 w7 w8 w9 …

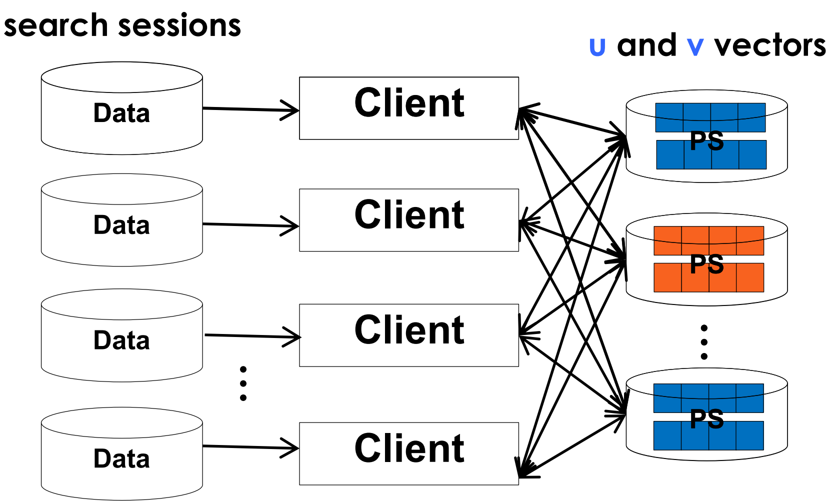
w3 w4 w6 w7

w3 w4 w6 w7

  - Updates involve vector multiply-accumulates (**v**+=α**u**, **u**+=α**v**, **v**+=β**u**, **u**+=β**v**), with α, β determined by (**u**•**v**, **u**•**v**).

Distributed Search2Vec

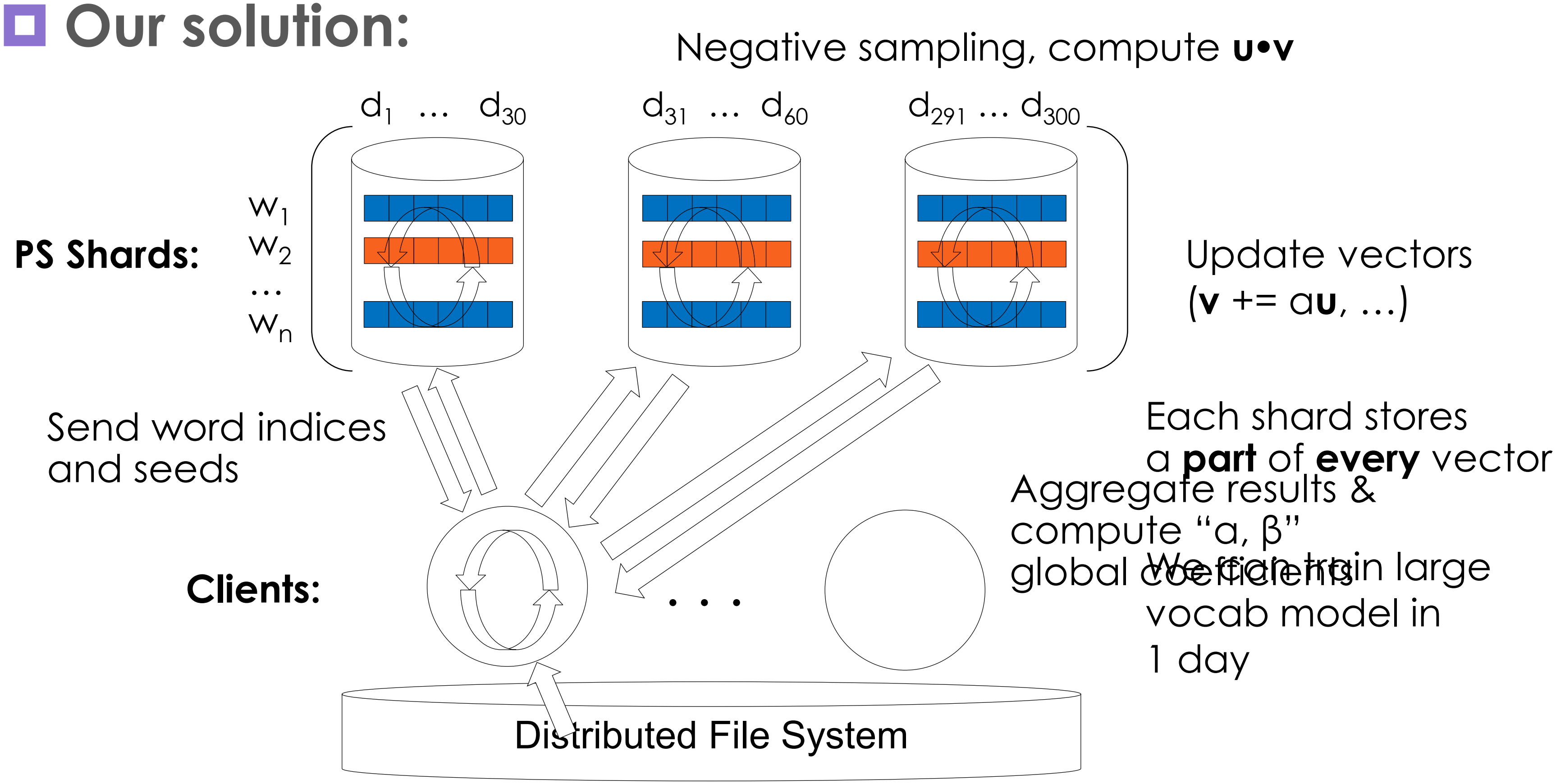
- **Parameter Server (PS)** - distributed in-memory store for model parameters (**vectors**), supports: **GET**, **PUT**



- 1 Client:
  - Take a mini-batch of data (e.g. 200 sessions)
  - **PS GET:** **v** vectors for each word from mini-batch and **u** vectors for **neighbors** and **random negatives**
  - Client calculates gradient updates for all **v** and **u**
  - **PS PUT:** updates **v** and **u** vectors in key-value store (**no locks**)

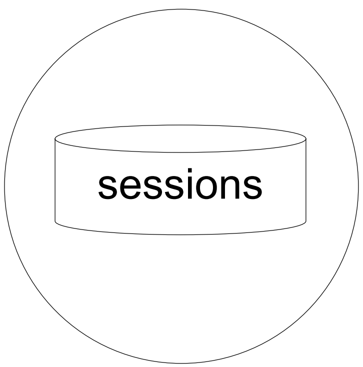
Distributed Search2Vec

- **Our solution:**



Distributed Search2Vec

client



1. send word indices and seed

waits ….

3. aggregate partial dot products and compute α, β weights

$$\alpha_{full} = \alpha_{1-30} + \alpha_{31-60} + \ldots + \alpha_{371-300}$$

$$\beta_{full} = \beta_{1-30} + \beta_{31-60} + \ldots + \beta_{371-300}$$

4. sends weights and seeds (again)

next mini-batch …

2. negative sampling + calculate partial dot products **u•v**

$$\alpha_{1-30} = \sum_{i=1}^{30} v_i u_i^{pos}$$

$$\beta_{1-30} = \sum_{i=1}^{30} v_i u_i^{neg}$$

5. update partial vectors **v** += α**u**, …

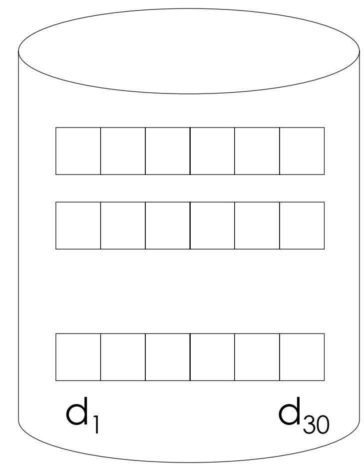
$$\mathbf{v}^{new}_{1-30} = \mathbf{v}_{1-30} + \eta \cdot (1 - \sigma(\alpha_{full})) \cdot \mathbf{u}^{pos}_{1-30}$$

$$\mathbf{v}^{new}_{1-30} = \mathbf{v}_{1-30} - \eta \cdot \sigma(\beta_{full}) \cdot \mathbf{u}^{neg}_{1-30}$$

$$\mathbf{u}^{pos}_{1-30} = \mathbf{u}^{pos}_{1-30} + \eta \cdot (1 - \sigma(\alpha_{full})) \cdot \mathbf{v}_{1-30}$$

$$\mathbf{u}^{neg}_{1-30} = \mathbf{u}^{neg}_{1-30} - \eta \cdot \sigma(\beta_{full}) \cdot \mathbf{v}_{1-30}$$

PS shard



Distributed S2V – A/B test

More vectors: ~300M query & ad vectors

- **Control**: prod + 1 machine s2v
- **Bucket**: prod + 1 machine s2v + distributed s2v

Bucket	Query Coverage	Auction Depth	CTR	Click Yield	Revenue per Search
distributed search2vec	**+2.44%**	**+2.39%**	**+0.2%**	**+1.81%**	**+9.39%**

Cold-start ad vectors

2. **cold ads**

How to generate vectors for new ads?

Title	Ancestry DNA Testing
Description	Learn More About Yourself & Your Family History.
Display URL	23andme.com/AncestryDNATesting
Bid Term	dna_testing

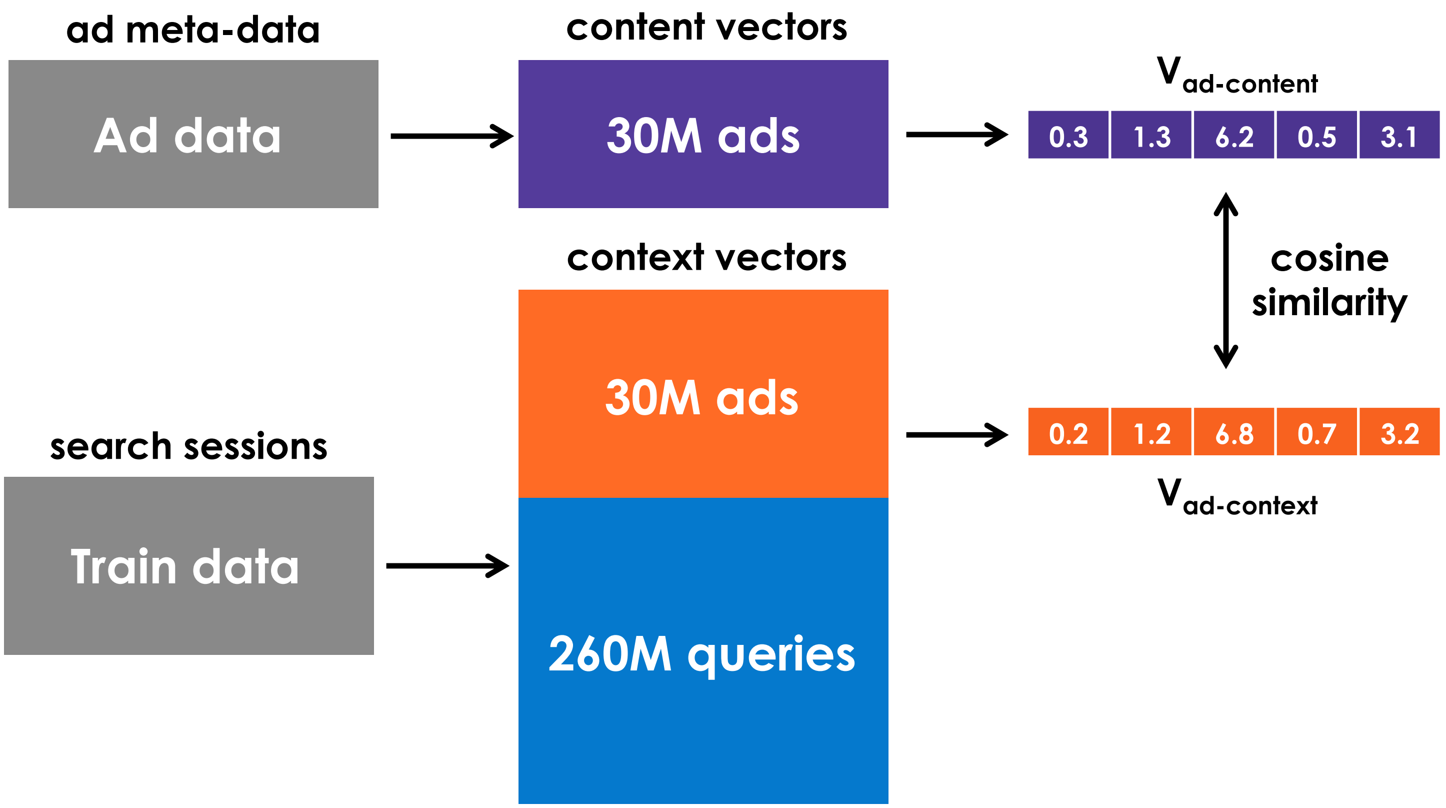
Cold-start ad vectors

source	n-gram	has vector	similarity to bid term
bidterm	dna_testing	YES	1
title	ancestry	YES	0.66
title	dna	YES	0.76
title	testing	YES	0.24
title	ancestry_dna	NO	
title	dna_testing	YES	1
title	ancestry_dna_testing	YES	0.87
description	learn	YES	0.11
description	more	YES	0.03
description	learn_more	NO	
description	about	YES	0.08
description	family_history	YES	0.62
description	your_family	YES	0.37

$$v_{ad\_content} = \underset{i}{\square}\, v_p(i)$$

Cold-start ad vectors

Offline Evaluation



Cold-start ad vectors

- **Offline Evaluation**
  - $\mathbf{V}_{\textbf{ad-context}}$: ad vectors learned from sessions
  - $\mathbf{V}_{\textbf{ad-content}}$: ad vectors formed from content
  - **sim** : average cosine sim. between $\mathbf{V}_{\textbf{ad-context}}$ and $\mathbf{V}_{\textbf{ad-content}}$
  - High **sim** tells us we came close to the "ground truth"

method	average	std
words	0.574	0.059
phrases	0.665	0.067
CRF phrases	0.604	0.075
bid term only	0.731	0.128
anchor phrases	**0.792**	**0.077**

Cold-start ad vectors – A/B tests

More ad vectors: additional 50M ad vectors

- **Control**: prod + distributed s2v
- **Bucket**: prod + distributed s2v + cold ad vectors

Bucket	Query Coverage	Auction Depth	CTR	Click Yield	Revenue per Search
Cold Start Ad Vectors	**+7.05%**	**+4.36%**	**-0.6%**	**+3.96%**	**+9.83%**

Our system today

- **Tail queries A/B tests** – still to come
- **search2vec today:**
  - top BROAD match algorithm
  - 30%+ of all BROAD match impressions
- **Read more about it at:** yahooresearch.tumblr.com

https://yahooresearch.tumblr.com/post/146257394201/science-powering-product-large-scale-query-to-ad

Vectors for Research Purposes

- 8M query vectors + 4K <query, query, grade> data available
- Webscope program: http://webscope.sandbox.yahoo.com/catalog.php?datatype=l&did=73
- Comparison to word2vec on query rewriting task:

Method	oAUC	Macro NDCG@5
word2vec	0.817	0.929
search2vec	**0.880**	**0.959**

Thank You!

- **Questions?**

Tail query vectors

3. **tail queries**

How to generate vectors for tail queries?

How to do online matching and leverage search2vec?

- Build an index for online matching
- Leverage head queries and form documents from their search2vec rewrites(gives us semantic expansions)
- For a new query: textual match against document, retrieve vector of the top result

Tail query vectors

3. **tail queries**

Step 1: find top K = 10 queries for each head query from the vocabulary

query	expansions	score
	free stock tickers	0.763
	stock ticker app	0.760
stock pro	best real time stock apps	0.757
	best stock tracker app	0.741
	free stock apps	0.732

Tail query vectors

3. **tail queries**

Step 2: form query documents (flatten)

id	document
scuba_diving_gear	scuba diving equipment diving gear scuba equipment scuba gear scuba shop
bread_machines	bread maker bread machines cusinart bread maker bread machine reviews bread machine recipes
met_opera_ny	met opera address met opera nyc metropolitan opera house new york city met opra metropolitan opera in nyc
stock_pro	free stock tickers stock ticker app best real time stock apps best stock tracker app free stock apps

Tail query vectors

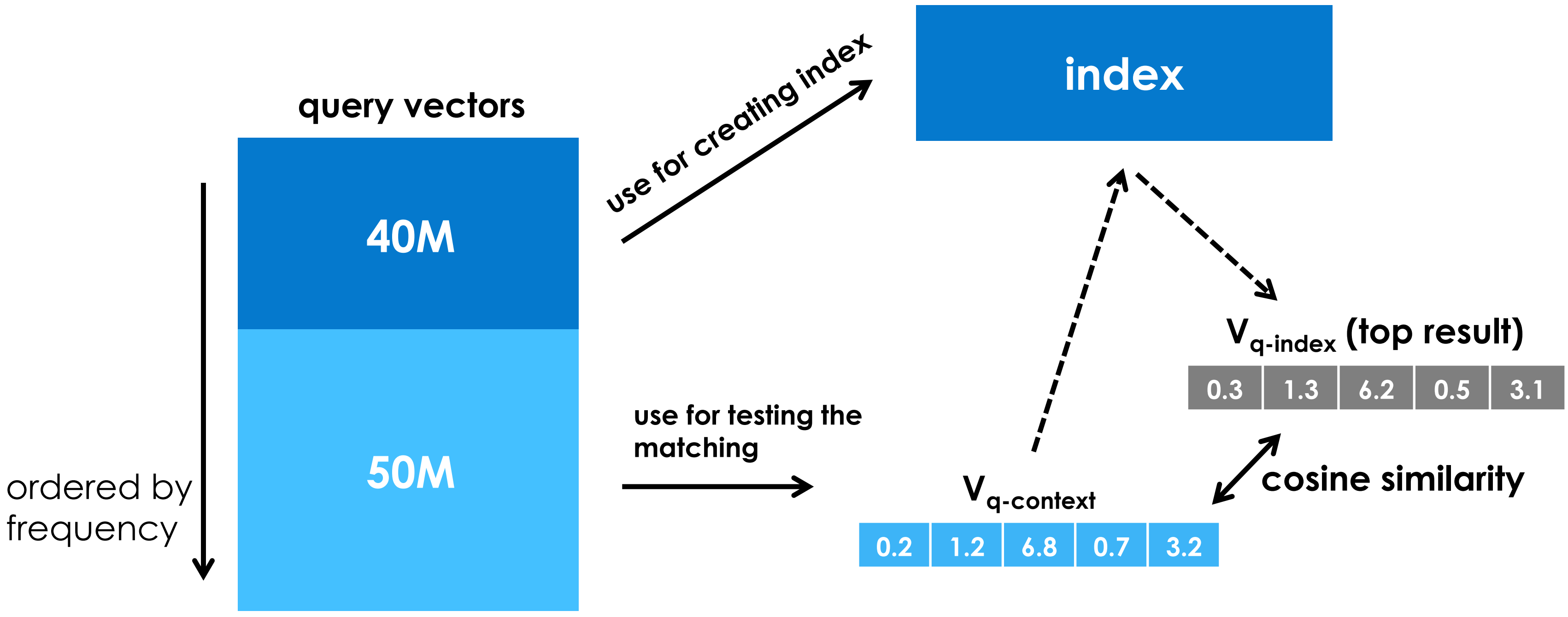
3. tail queries

Step 3: invert index for fast matching

input query	top result	matched document
metropolitan opera house that is in new york city	new_york_city_opera	**metropolitan opera** tosca ny city opera **metropolitan opera** promo code **new york city opera new york city opera** company **metropolitan opera** website **metropolitan opera house** lincoln center met **opera new york metropolitan opera** dress code **metropolitan opera** discount tickets
malware bytes free edition software download	free_antimalware_software	**free malware software download malware bytes download** free malwarebytes **downloads** anti **malware malware** anti **malware** antivirus **free download** norman **malware** free anti malwarebytes **free** edition free antimalware **software**
what is the best stock ticker trading app in appstore?	stock_pro	free **stock tickers stock ticker app best stock** chart **best** real time **stock** charts **best stock tracker app** free **stock apps stock** tracker software good stocks to day trade free **stock** market **ticker stock** pro

Tail query vectors

Evaluation



Tail query vectors - Evaluation

- **Offline Evaluation**
  - $\mathbf{V}_{\mathbf{q\text{-}context}}$: query vectors learned from sessions (50M)
  - $\mathbf{V}_{\mathbf{q\text{-}index}}$: query vectors formed by leveraging index (50M)
  - **sim** : average cosine sim. between $\mathbf{V}_{\mathbf{q\text{-}context}}$ and $\mathbf{V}_{\mathbf{q\text{-}index}}$
  - High **sim** tells us we came close to the "ground truth"

method	average	std
words	0.452	0.101
phrases	0.574	0.120
CRF phrases	0.514	0.119
elastic co-occurred queries K=10	0.621	0.084
elastic s2v K=10	**0.717**	**0.091**