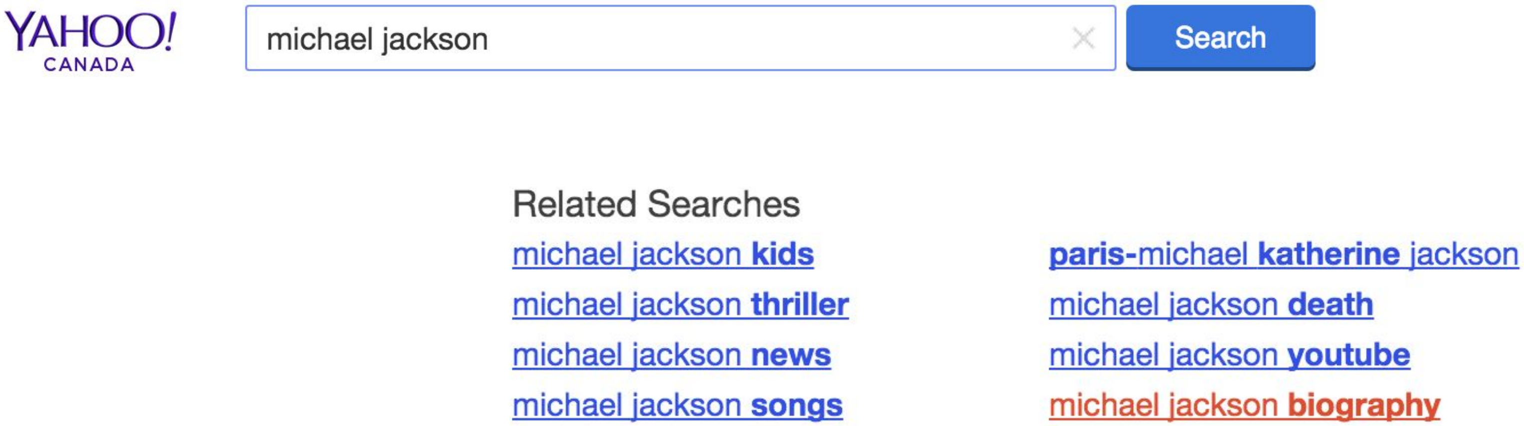
A Hierarchical Recurrent Encoder-Decoder for Context-Aware Generative Query Suggestion

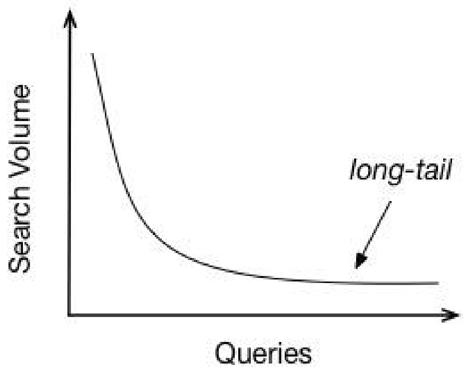
Alessandro Sordoni, Yoshua Bengio, Puya-Hossein Vahabi
Christina Lioma, Jakob Simonsen, Jian-Yun Nie

Query Suggestions



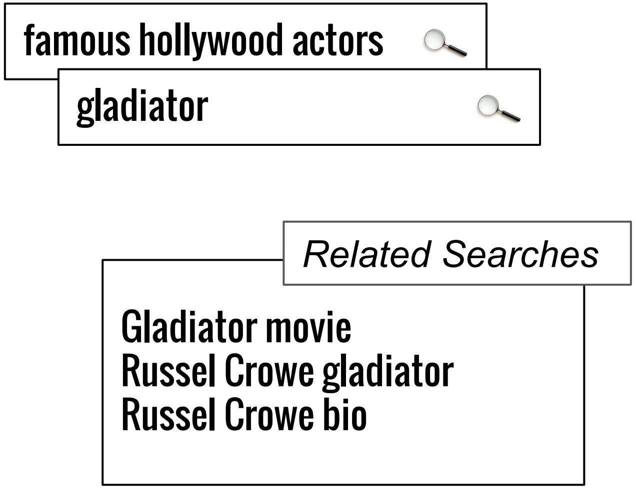
Some Desiderata of a Suggestion System

Long-tail, i.e. find suggestions for rare queries, where co-occurrence systems may fail due to data sparsity.



Some Desiderata of a Suggestion System

Context-aware - i.e. being able to account for the recent user query history, moving beyond the most recent query.



Some Desiderata of a Suggestion System

Generative - i.e. being able of producing synthetic suggestions that may not exist in the training data.

Query Suggestion SOTA

- Query-Flow Graph and Term-Query Graph [Bonci et al. 2008, Vahabi et al. 2012]
  - Robust to long-tail queries but computationally complex
- Context-awareness by VMM models [He et al. 2009, Cao et al. 2008]
  - Sparsity issues and not robust to long-tail queries
- Learning to rank by featurizing query context [Shokhoui et al. 2013, Ozertem et al. 2012]
  - Order of queries / words in the queries is often lost
- Synthetic queries by template-based approaches [Szpektor et al. 2011, Jain et al. 2012]

Our work

- Novel Recurrent Neural Network (RNN) for query suggestion.
- Key Properties :

1) *robust in the long-tail* - word-based approach
2) *context-aware* - can use an unlimited number of previous queries
3) *generative* - synthetic queries, sampled one word at the time

Word and Query Embeddings

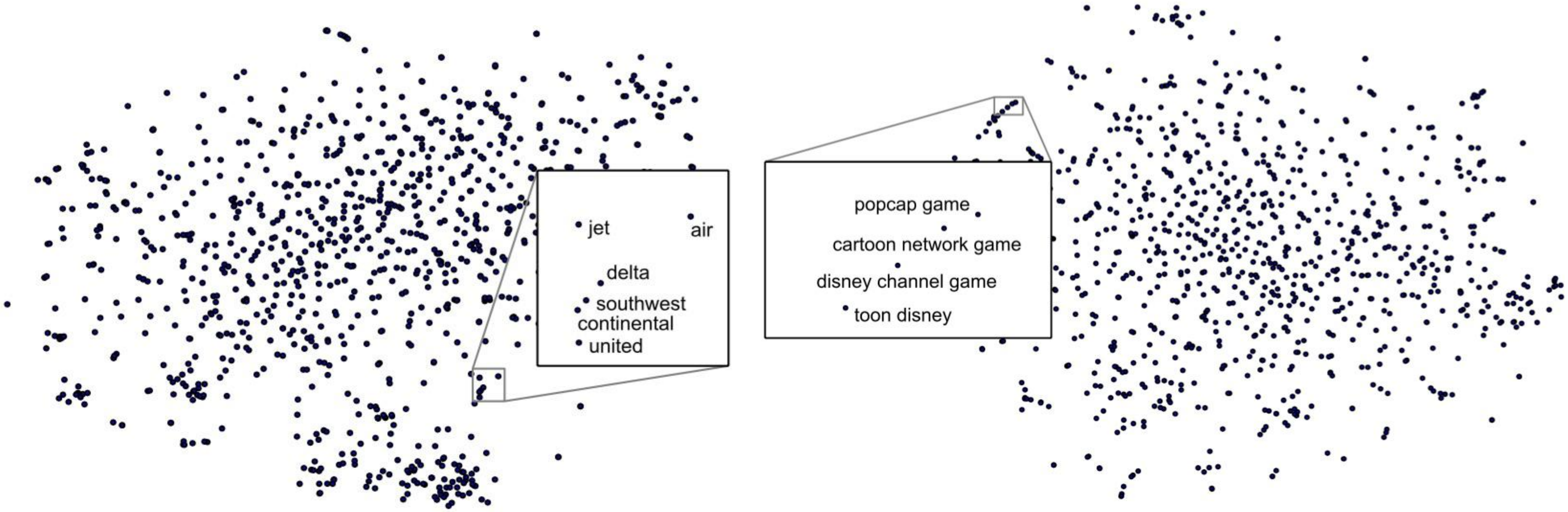
Learn vector representations for **words** and **queries** encoding their syntactic and semantic characteristics.

“game” = [0.1, 0.05, -0.3, … , 1.1]

“cartoon network game” = [0.35, 0.15, -0.12, … , 1.3]

“Similar” queries associated to “near” vectors.

Word and Query Embeddings

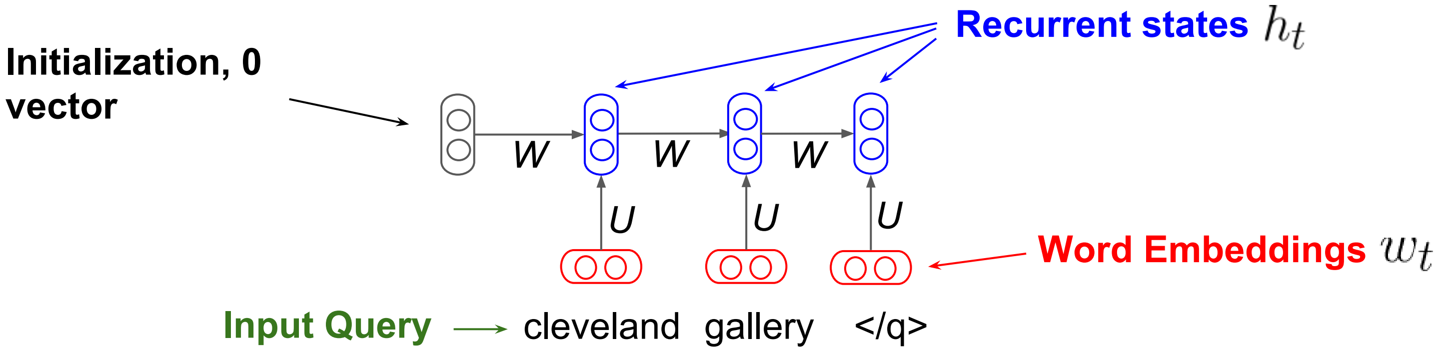


word space

query space

Recurrent Neural Networks (RNNs)

- RNNs model arbitrary time sequences, such as a sequence of query words.

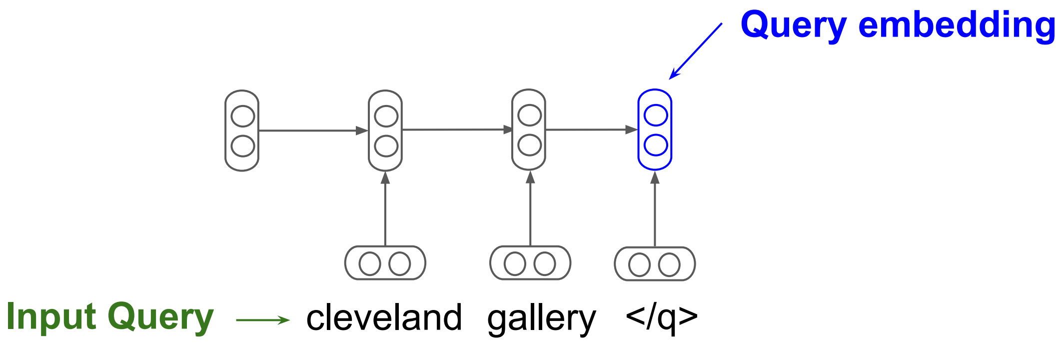


$$h_t = \tanh(W h_{t-1} + U w_t)$$

- The weight matrices *W* and *U* are fixed throughout the timesteps.

RNN encoder

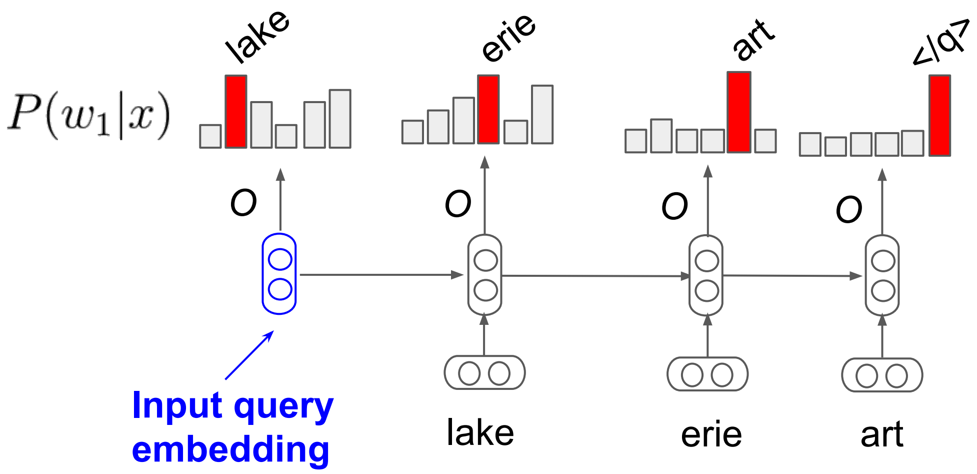
- Aggregates word embeddings
- The last recurrent state is used as the *query embedding*.



- The query embedding is sensitive to the order of words in the query !

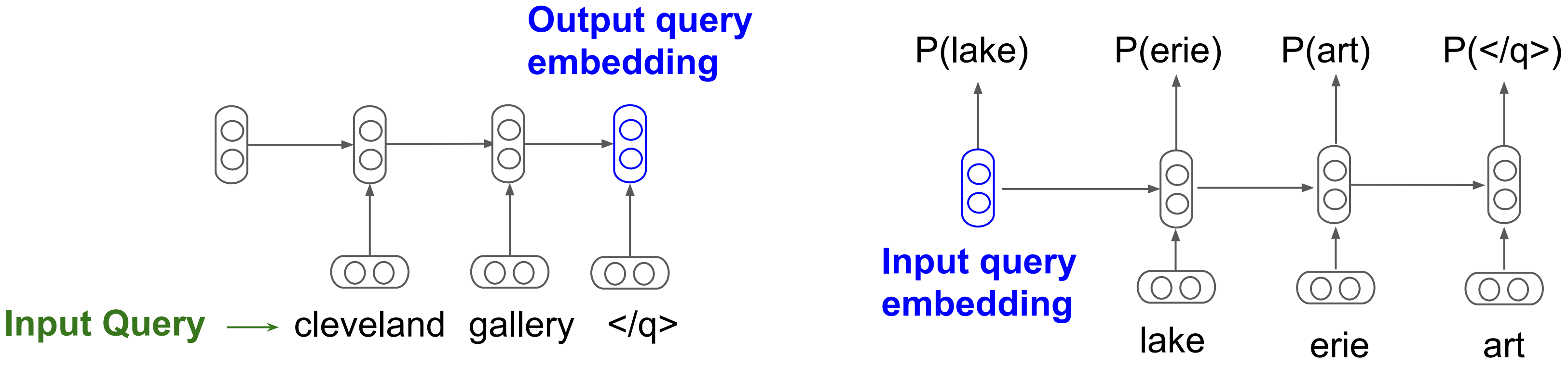
RNN decoder

- Recurrent states are used to predict the next word in the output query.
- Probabilistic mapping from query embeddings to textual queries, $P(Q|x)$



$$P(w_{t+1}|w_t, \ldots, w_1, x) = \mathrm{softmax}(Oh_t + b) \quad O \in \mathbb{R}^{V \times h}$$

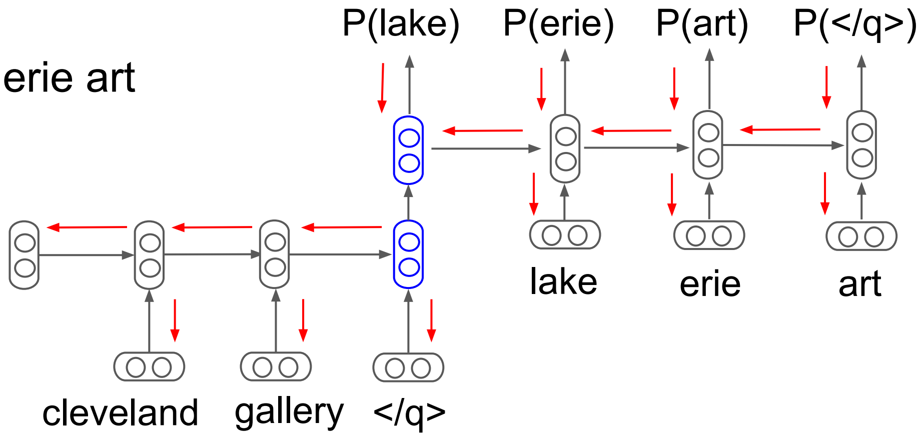
RNN encoder and RNN decoder



Recurrent Encoder-Decoder (RED)

- A RNN encoder-decoder (RED) learns a probability distribution over the next-query in the session given the previous one.

S = cleveland gallery → lake erie art



- Backprop Training: $L = \log P(Q_{t+1}|Q_t) = \sum_{w_n \in Q_{t+1}} \log P(w_n|w_{<n}, Q_t)$

Problem with RED

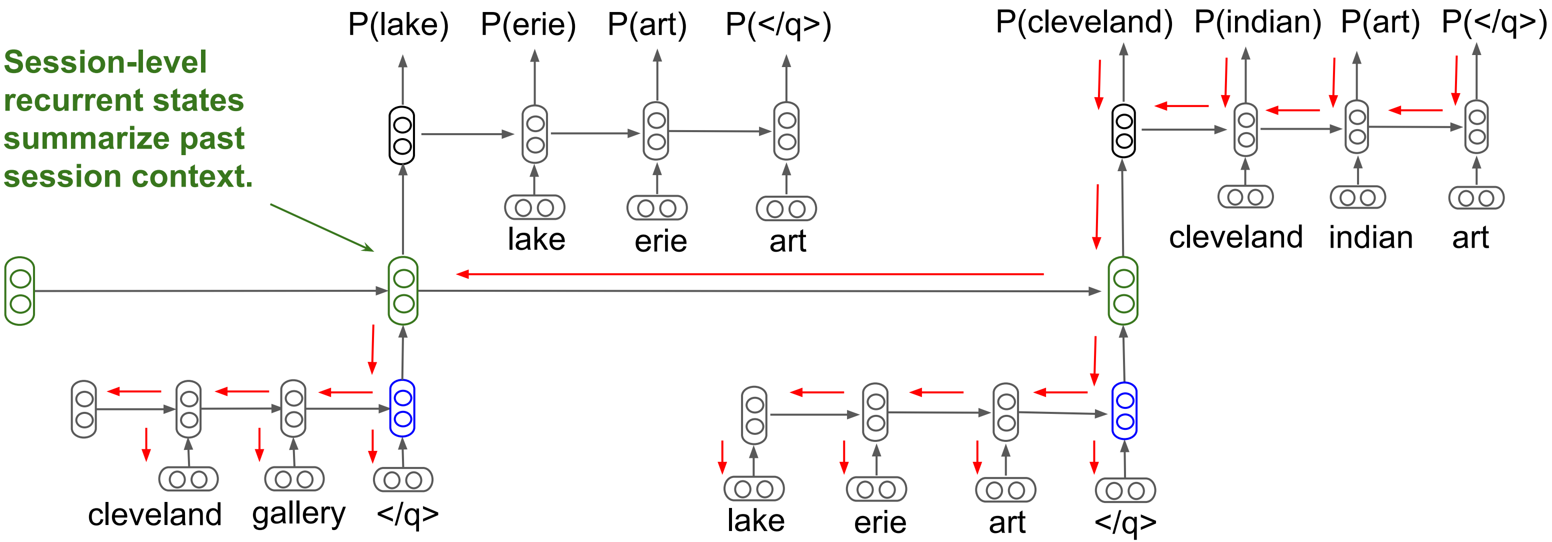
- The RED model is purely *pairwise*, while we know that sessions are composed by several queries that needs to be considered as context.

S = cleveland gallery → lake erie art → cleveland indian art

Hierarchical Recurrent Encoder Decoder (HRED)

- Use an additional RNN to model the sequences of queries in a session.

cleveland gallery → lake erie art → cleveland indian art



Example synthetic suggestions

Context	Synthetic Suggestions
ace series drive	ace hardware ace hard drive hp officejet drive ace hardware series
cleveland gallery → lake erie art	cleveland indian art lake erie art gallery lake erie picture gallery sandusky ohio art gallery

Experiments

Experimental Setting

- Experimental setup based on (Shokohui, 2013; Mitra, 2015)
- How well the suggestion model can predict the next query in the session ?
- AOL query log, temporally separated background, train, validation and test sets

	# of Sessions
Background	1.7 M
Train	435 K
Validation	170 K
Test	230 K

Learning to rank the next query

- Context-aware next-query prediction as a learning-to-rank task:

Context: cleveland gallery → lake erie art → Target: cleveland indian art

- 20 Negative, out-of-context candidates by using adjacency counts **(ADJ)**

lake erie art → { lake erie photography, lake erie gallery }

- Rerank candidates using a LambdaMART model.

20 Features

Non-contextual features

Session length, candidate frequency

Contextual features

QVMM model [He et al. 2009], N-gram features from [Mitra et al. 2015]

Pairwise features, computed between last context query and each candidate

ADJ counts, Levensthein and n-gram distance

HRED

Log-likelihood of each candidate given the session context

Results - Overall

HRED features improve significantly over pairwise ADJ model and the context-aware baseline ranker.

Method	MRR	Δ%
ADJ	0.5334	-
Baseline Ranker	0.5563	*+4.3%*
+ HRED	**0.5749**	*+7.8%/+3.3%*

Impact of Session Length

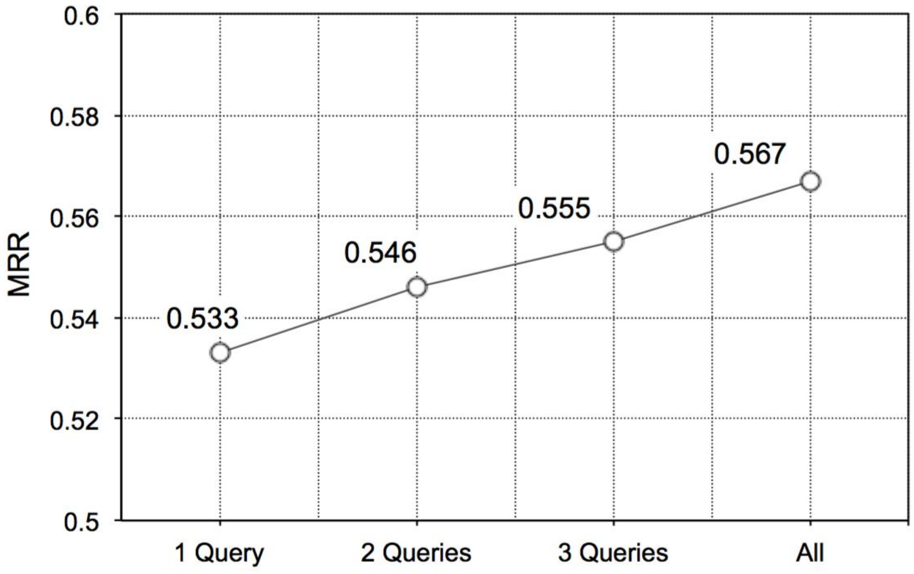
Short (2 queries)
Medium (3 - 5 queries)
Long sessions (> 5 queries)

Biggest improvements of HRED on medium and long sessions.

Impact of the Context Length

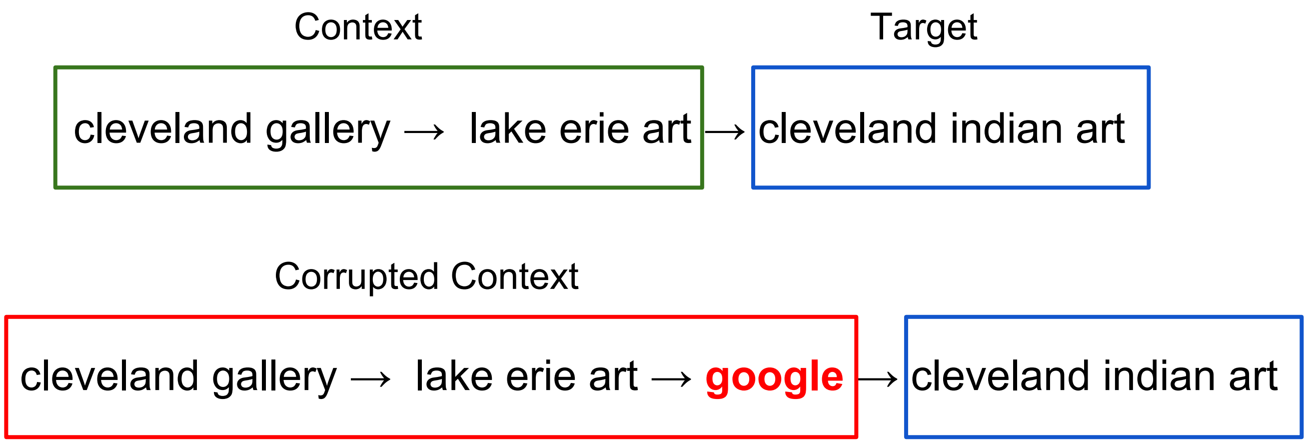
Artificially vary the number of context queries considered by HRED on long sessions

HRED can effectively exploit more than 3 queries in the context, thus capturing long-range dependencies.



Robust Prediction

- Context-aware methods should be robust to noise in the session.
- Randomly corrupt context by inserting “noisy” queries (top-100 most frequent queries in the query log) at a random position.



Robust Prediction Results

ADJ suffer a significant drop in MRR on corrupted sessions.

Relative improvements of HRED are ~3x higher compared to the original setting denoting robustness to the noisy query.

Original Sessions

Method	MRR	Δ%
ADJ	0.5334	-
Baseline Ranker	0.5563	*+4.3%*
+ HRED	**0.5749**	*+7.8%/+3.3%*

Corrupted Sessions

Method	MRR	Δ%
ADJ	0.4507	-
Baseline Ranker	0.4831	*+7,2%*
+ HRED	**0.5309**	*+17,8%/+9.9%*

Long Tail Prediction

Last query in the context is a long-tail query, unseen in the training data.

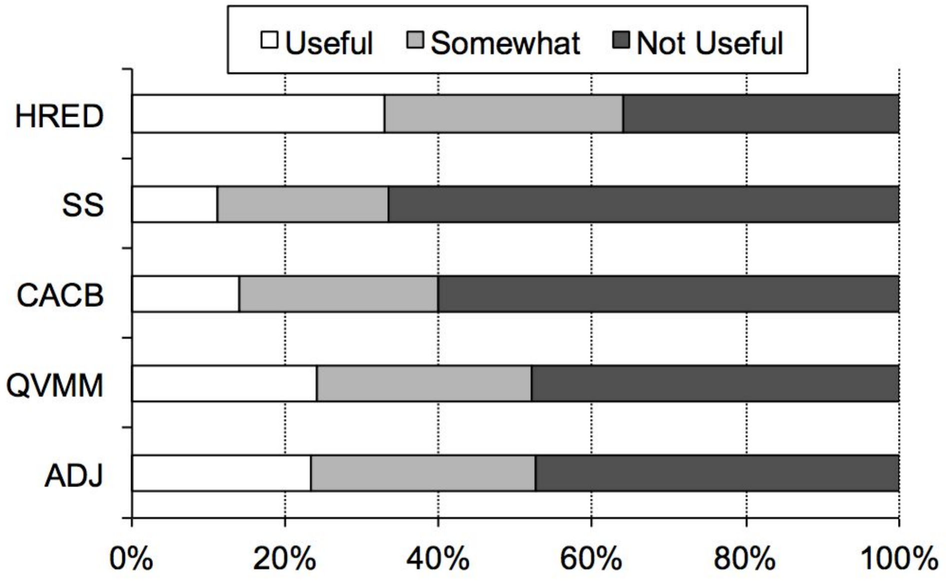
Method	MRR	Δ%
ADJ	0.3830	-
Baseline Ranker	0.6788	*+77.2%*
+ HRED	**0.7112**	*+85.3%/+5.6%*

Human Eval

50 queries from TREC Web Track 2012 with artificial context

5 Raters judge the top-5 suggestions for each method

HRED was used in generation mode, beam-sampling size 25



Summary of Contributions

- A query log session language model based on a RNN architecture.
- A hierarchical architecture to model long-range session context.
- First application of RNNs to query suggestion.
- Improve performance on MRR up to 3.3% overall and up to 10% on long sessions where context matters the most.
- Improve MRR on noisy sessions up to 9.9%.
- Improve MRR on sessions up to 5.6% in the long-tail setting.

Co-occurrence Suggestion System

1. Count session level pairwise co-occurrences.
2. Most co-occurring queries as suggestions.

dys → </S>

cleveland gallery → lake erie art → </S>

(lake erie art, cleveland gallery) = 1
(</S> , dys) = 1
(</S> , lake erie art) = 1

Co-occurrence Suggestion System

1. Count session level pairwise co-occurrences.
2. Most co-occurring queries as suggestions.

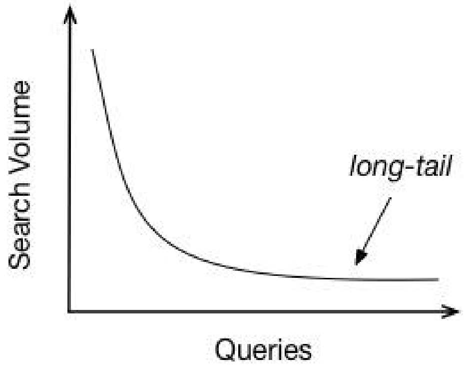
dys → </S>

cleveland gallery → lake erie art → </S>

(lake erie art, cleveland gallery) = 1
(</S> , dys) = 1
(</S> , lake erie art) = 1

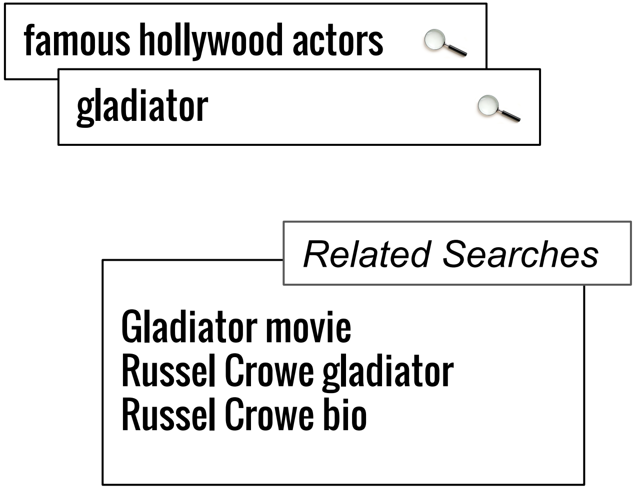
Some Desiderata of a Suggestion System

Long-tail, i.e. find suggestions for rare queries, where co-occurrence systems may fail due to data sparsity.



Some Desiderata of a Suggestion System

Context-aware - i.e. being able to account for the recent user query history, moving beyond the most recent query.



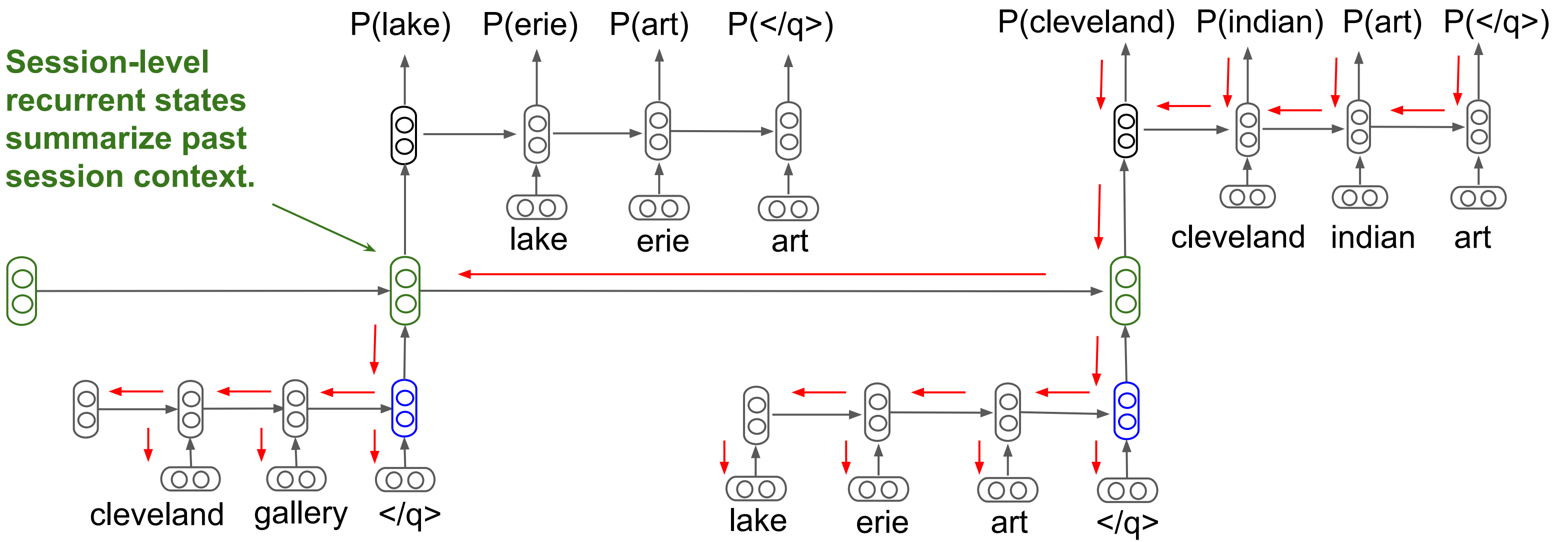
Some Desiderata of a Suggestion System

Generative - i.e. being able of producing synthetic suggestions that may not exist in the training data.

Hierarchical Recurrent Encoder Decoder (HRED)

- Use an additional RNN to model the sequences of queries in a session.

cleveland gallery → lake erie art → cleveland indian art



Hierarchical Recurrent Encoder Decoder (HRED)

- Training: given a query session S, maximize the likelihood of the session computed by HRED using gradient descent:

$$L(S) = \sum_{m=1}^{|S|} \log P(Q_m|Q_{1:m-1}) = \sum_{m=1}^{|S|} \sum_{n=1}^{|Q_m|} \log P(w_{m,n}|w_{m,1:n-1}, Q_{1:m-1})$$

- Suggestion: decode the most probable query given session context

$$Q^* = \arg\max_{Q} P(Q|Q_{1:m})$$

- Rescoring: compute the likelihood of a suggestion given the context

$$s(Q) = P(Q|Q_{1:m})$$