Generic Intent Representation in Web Search

Hongfei Zhang, Xia Song, Chenyan Xiong, Corby Rosset,
Paul N. Bennett, Nick Craswell, and Saurabh Tiwary
Microsoft AI & Research
{honzha, xiaso, cxiong, corosset, pauben, nickcr, satiwary}@microsoft.com

ABSTRACT

This paper presents GEneric iNtent Encoder (GEN Encoder) which learns a distributed representation space for user intent in search. Leveraging large scale user clicks from Bing search logs as weak supervision of user intent, GEN Encoder learns to map queries with shared clicks into similar embeddings end-to-end and then fine-tunes on multiple paraphrase tasks. Experimental results on an intrinsic evaluation task – query intent similarity modeling – demonstrate GEN Encoder's robust and significant advantages over previous representation methods. Ablation studies reveal the crucial role of learning from implicit user feedback in representing user intent and the contributions of multi-task learning in representation generality. We also demonstrate that GEN Encoder alleviates the sparsity of tail search traffic and cuts down half of the unseen queries by using an efficient approximate nearest neighbor search to effectively identify previous queries with the same search intent. Finally, we demonstrate distances between GEN encodings reflect certain information seeking behaviors in search sessions.

KEYWORDS

Generic Intent Representation, Query Embedding, User Intent.

ACM Reference Format:
Hongfei Zhang, Xia Song, Chenyan Xiong, Corby Rosset, Paul N. Bennett, Nick Craswell, and Saurabh Tiwary. 2019. Generic Intent Representation in Web Search. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, San Francisco, USA, 10 pages. https://doi.org/10.1145/3331184.3331198

1 INTRODUCTION

User intent understanding plays a fundamental role in modern information retrieval. A better understanding of "what a user wants" helps search engines return more relevant documents, suggest more useful queries, and provide more precise answers. Intent understanding is also challenging: two queries with the same intent may have no term overlap (e.g. "cheap cars", "low-priced autos") while two queries with slight variations may have completely different meanings (e.g. "horse racing", "racing horses"). A brittle bag-of-words style of query representation faces such challenges as vocabulary mismatch and ambiguity [7].

Distributed representations (embeddings) provide a path to address these challenges and have been increasingly explored with the rapid development of neural methods. Representing text as continuous embeddings enables *softer matches* than term overlap. The embeddings also incorporate *additional information*. For example, word2vec maps semantically similar words together by learning they often appear in similar surrounding texts [22]. The embeddings also support *richer compositionality* than bag-of-words using neural networks. For example, ELMo and BERT use deep neural architectures to provide more context-aware embeddings [9, 27].

Nevertheless, embeddings learned from surrounding texts do not necessarily capture user intents in search queries. The query "Harvard student housing" has similar word2vec embeddings with "Cornell student housing", but its intent — what a user would find relevant in the search results — is closer to "Cambridge dorm MA" [36]. Previous work in neural information retrieval demonstrated that these semantic-oriented embeddings often cause topic drift in search [28, 40]. It is often necessary to train task-specific embeddings [25, 36, 41], while it is not quit clear how to construct generic representations for search intents.

This work presents GEneric iNtent (GEN) Encoder, which learns a distributed representation space for user intent from user feedback in search. GEN Encoder uses a character aware recurrent architecture and a two phase learning strategy. It first utilizes user clicks as weak supervision of search intent and learns to encode queries which lead to clicks on the same documents (co-click queries) near each other in its representation space. Click signals are available at scale in Bing search logs and thus provide sufficient signals to learn GEN Encoder end-to-end. Then, GEN Encoder employs multi-task learning on paraphrase classification tasks to improve its generality.

We present an intrinsic evaluation – query intent similarity modeling – to evaluate the quality of intent representations by their ability to group queries with similar intents together. Three datasets at different difficulties were constructed with query pairs sampled from search sessions and assessor labels of their intent similarities. The results demonstrate GEN Encoder's robust advantages over previous representation methods including, discrete bag-of-words, embedding-based bag-of-words, relevance-based embeddings, and generic encoders designed for modeling language semantics.

Thorough ablation studies reveal the critical role of user clicks in learning intent representations. Simple encoder architectures trained from co-clicks nearly achieve the performances of previous state-of-the-art text encoders, while complex encoder architectures without co-click signals fail to outperform TF-IDF. Thus, the source of supervision signals are a more important first consideration than neural architecture. Results also demonstrate the importance of multi-task learning in building a generic representation space and the effectiveness of the GEN Encoder architecture.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGIR '19, July 21–25, 2019, Paris, France
© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331198

GEN Encoder has many applications in search. We first demonstrate that it alleviates the sparsity of tail search traffic. Search intelligence highly depends on learning from previous observations of user behaviors on the query [1, 16, 19]. However, the long-tail nature of search means many queries are rarely, if ever, observed. GEN Encoder reduces this sparsity by identifying previously observed queries with the same search intent, using approximate nearest neighbor (ANN) search in the continuous space. Our experiments with an ANN index of 700 million queries demonstrate that this can be achieved with high coverage, accuracy, and practical latency: ANN search returns neighbors for the majority of queries and finds more than one neighbor with the exact same intent, at the speed of 10 milliseconds per query. Incorporating the evidence from these semantically similar queries reduces the fraction of unseen queries from 38% to 19%, cutting down half of the long tail sparsity.

Finally, we demonstrate emergent behavior where distance in GEN encodings separates common types of query reformulation behavior in a search session: topic change, exploration, specification, and paraphrase. Our quantitative study found that the distances between the GEN encoding of query reformulations are nicely separated to a bi-modal distribution, with different ranges corresponding to different information-seeking relationships between the queries. While not designed explicitly for this purpose, the evidence suggests that GEN Encoder may have future applications in analyzing and understanding user behaviors in sessions.

A generic distributed representation space capturing user intent—plus efficient approximate nearest neighbor retrieval at scale—has the potential to fundamentally change the way search is conducted. GEN Encoder is publicly available to facilitate more future exploration in this direction[1].

2 RELATED WORK

In user intent analysis, Broder's influential study grouped intents ("the need behind the query") into predefined categories: Informational, Navigational, and Transactional, based on search tasks [3]. Another dimension in modeling intent is to categorize them into predefined or automatically constructed taxonomies [4, 37]. The classification is often done by machine learning models, using information from Wikipedia [15], query-click bipartite graph [19], and pseudo relevance feedback [31]. Query intent classification is a standard component in search engines and plays a crucial role in vertical search [19] and sponsored search [4].

Many approaches have been developed to improve the bag-of-words query representation. The weights of query terms can be better estimated by inverse document frequency (IDF) [7] and signals from query logs [2]. The sequential dependency model [21] incorporated n-grams. Query expansion techniques enrich queries with related terms from relevance feedback [29], pseudo relevance feedback (PRF) [18], or external knowledge graphs [35]. These techniques are the building blocks of modern retrieval systems.

Embedding techniques provide many new opportunities to represent queries and model user intent in a distributed representation space. Zheng and Callan utilize the semantics of word embeddings in query term weighting [42]. Zamani and Croft use word embeddings to build a smooth language model [38] and derived the theoretical framework to combine word embeddings into query embeddings [39]. Guo et al. incorporate the soft match between query and document in their word embedding for ad hoc ranking [13].

Later, it was found that word embeddings trained from surrounding contexts are not as effective in modeling user intent, information needs, or relevancy [36, 40]. Word2vec embeddings align word pairs reflecting different user intents together [36] and cause topic drift [28]. It is preferred to learn a distributed representation directly from information retrieval tasks. Nalisnick et al. compared the effectiveness of word embeddings trained on search queries versus trained on documents in ad hoc retrieval [24]. Diaz et al. demonstrated that word embeddings trained on each query's PRF documents are more effective in query expansion [10]. Hamed and Croft trained relevance-base word embeddings from query and PRF term pairs, which are more effective in embedding-based language model [40]. In ad hoc ranking, neural models are more effective if using embeddings learned end-to-end from relevance labels than using embeddings trained from surrounding contexts, showing the different demands of embeddings in search [8, 25, 36, 41].

Effective as they are, end-to-end learned embeddings are not always feasible: many tasks do not have sufficient training data and not every task is supervised. This motivated the development of *universal representations*, which aims to provide meaningful and generalizable distributed representations for many language processing tasks. The Universal Sentence Encoder learns a universally effective encoder using multi-task learning on many language modeling tasks [6]. Contextual embeddings, such as ELMo [27] and BERT [9], learn deep neural networks from large corpora and provide contextualized embeddings on sentences and longer texts. These universal representations effectively capture general language semantics and have shown promising results on many language processing tasks, though their impact on search intent representation is unclear.

This paper brings in recent neural techniques to model search queries (§3.1) and is the first to leverage co-click weak supervisions and multi-task learning to learn distributed intent representations (§3.2). We create the first intrinsic evaluation for query intent representations with scaled intent similarity judgments (§4.1). Evaluation results and analyses demonstrate the effectiveness of our method and the contributions of learning from user clicks, multi-tasks, and the encoder architecture (§4.2). To illustrate the general impact in search, we demonstrate how GEN Encoder alleviates the sparsity in tail traffic (§5), and how it helps understanding information seeking behaviors in sessions (§6)

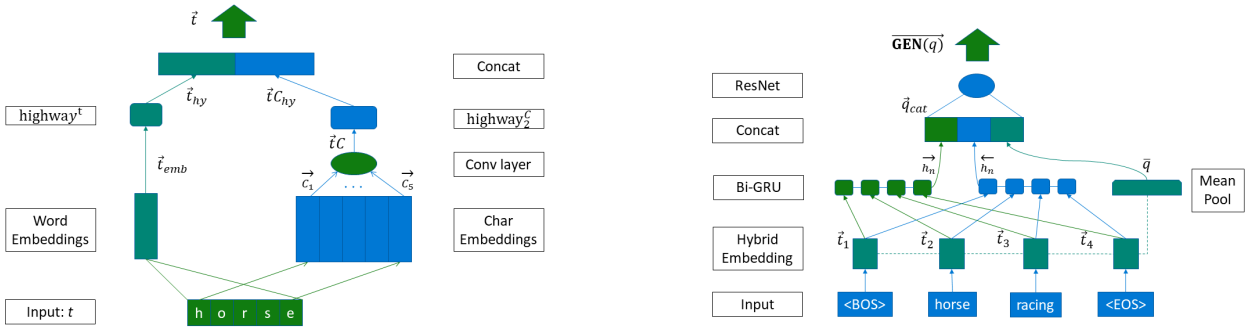
3 GENERIC INTENT ENCODER

GEN Encoder aims to learn a generic representation space that captures user intent in web search. This section first describes its architecture and then its learning strategy.

3.1 Architecture

As shown in Figure 1, GEN Encoder embeds a query q to a continuous vector $\overrightarrow{\text{GEN}}(q)$ through three components: a *word embedding* that maps query words to continuous vectors (1a), a *character-aware embedding* that models rare words with morphology (1a), a *mix encoder* that composes word embeddings to query encoding (1b).

[1] Aka.ms/GenEncoder



(a) Word and Character Embedding

(b) Mix Encoder

Figure 1: GEN Encoder Architecture

Word Embedding maps words into a continuous space, which aligns words with similar intents, e.g. "housing" and "dorm", and separates words with different intents, e.g. "Harvard" and "Cornell".

The query term t first goes through a standard embedding layer:

$$t \xrightarrow{emb} \vec{t}_{emb},$$

which learns embeddings for all terms in the vocabulary. The embeddings are fed into a highway network for better model capacity [33]:

$$\begin{aligned}\vec{t}_{hy} &= \text{highway}^t(\vec{t}_{emb}; g^t_{hy}, p^t_{hy}) \\ &= g^t_{hy}(\vec{t}_{emb}) \times p^t_{hy}(\vec{t}_{emb}) + (1 - g^t_{hy}(\vec{t}_{emb})) \times \vec{t}_{emb}. \end{aligned} \tag{1}$$

It includes a non-linear projection:

$$p^t_{hy} = \text{relu}(w_{p^t_{hy}} \times \vec{t}_{emb}), \tag{2}$$

and a gate which controls the projection:

$$g^t_{hy}(\vec{t}_{emb}) = \text{sigmoid}(w_{g^t_{hy}} \times \vec{t}_{emb}). \tag{3}$$

Relu and sigmoid are activations; $w_{p^t_{hy}}$ and $w_{g^t_{hy}}$ are parameters.

Character-Aware Embedding. Search traffic follows a long tail distribution: many query terms only appear a few times or never before for reasons such as misspelling. To help alleviate out-of-vocabulary issues, we employ a convolutional neural network over character embeddings to help represent rare words [16, 17]: the character-aware embeddings of "retreval" and "retrieval" can be similar because of shared characters, but the word embedding of "retreval" may not be well trained due to its low frequency caused by spelling errors.

The characters $C = \{c_1, ..., c_j, ..., c_m\}$ of term t are embedded by an embedding layer, a Convolution layer, and a highway network.

$$c_j \xrightarrow{emb} \vec{c}_j, \tag{4}$$

$$\vec{t}_C = \text{max-pool}(\text{CNN}(c_1, ..., c_j, ..., c_m)), \tag{5}$$

$$\vec{t}_{C_{hy}} = \text{highway}^C_2(\vec{t}_C; g^C_{hy1}, p^C_{hy1}, g^C_{hy2}, p^C_{hy2}). \tag{6}$$

Eq. 4 embeds each character; Eq. 5 composes the embeddings of character n-grams and max-pools them to $\vec{t}_C$; Eq 6 is the same as Eq. 1 but has one more layer as the character vocabulary is smaller and can use more added capacity.

The word embedding (Eq. 1) and the character-aware embedding (Eq. 6) are concatenated to the final term embedding:

$$\vec{t} = \vec{t}_{hy} \frown \vec{t}_{C_{hy}}. \tag{7}$$

Mix Encoder composes the word embeddings $\{\vec{t}_1, ..., \vec{t}_i, ..., \vec{t}_n\}$ into the query encoding $\vec{\text{GEN}}(q)$. It combines a sequential encoder to handle order sensitive queries, e.g. "horse racing" and "racing horse", and a bag-of-words model to model order insensitive queries, e.g. "Cambridge MA" and "MA Cambridge".

The sequential encoder is a one-layer bi-directional GRU:

$$\{\overleftarrow{h_n}, \overrightarrow{h_n}\} = \text{Bi-GRU}(\vec{t}_1, ..., \vec{t}_i, ..., \vec{t}_n). \tag{8}$$

$\overleftarrow{h_n}$ and $\overrightarrow{h_n}$ are the last states of the two directions.

`GEN Encoder` combines the Bi-GRU with the average of query word embeddings (bag-of-words) using a residual layer:

$$\vec{\text{GEN}}(q) = \tanh(\vec{q}_{cat} + \text{relu}(w^q_{rs} \times \vec{q}_{cat})), \tag{9}$$

$$\vec{q}_{cat} = \overleftarrow{h_n} \frown \overrightarrow{h_n} \frown \frac{1}{n}\sum_i \vec{t}_i, \tag{10}$$

where w^q_{rs} is the parameter. $\vec{\text{GEN}}(q)$ is the encoding of q.

3.2 Learning

The distributed representations generated by `GEN Encoder` are determined by the information it learns from, which comes in two learning phases. The first leverages user clicks as weak supervision to capture user intent; the second uses a combination of paraphrase labels in a multi-task setting to improve generality.

Weakly Supervised Learning. Understanding user intents is difficult as they are not explicitly stated by the user. User clicks, however, have been widely used as implicit feedback of what a user wants [7]. The first learning phase follows this intuition and assumes that queries with similar user clicks have similar user intents. It trains `GEN Encoder` to produce similar embeddings for such co-click queries and distinguishes those without shared clicks.

Table 1: Statistics and Examples of the Three Intent Similarity Datasets.

Data (# Target Q)	Label	Count	Example Pair	
General (3,773)	Good	2,720	"evaluation of instant checkmate"	"review of instant checkmate"
	Fair	2,609	"explorer how to export bookmarks"	"how to export bookmarks"
	Bad	9,804	"fantasy football contract leagues"	"fantasy football league trophies"
Easy (2,864)	Good	14,180	"cream of mushroom soup recipe"	"cream mushroom soup recipe"
	Fair	7,094	"louisiana cdl manual"	"louisiana cdl training manual"
	Bad	23,027	"how dose yelp work"	" how long does yelp work"
Hard (385)	Good	193	"how to get rid of pimples"	"a fast way to get rid of pimples"
	Bad	233	"horse racing"	"racing horse"

Let (q, q^*) be a pair of co-click queries sampled from the search log, the loss function of the first phase is

$$l_{\text{co-click}} = \sum_q \frac{1}{1 + \exp(\cos(\vec{\texttt{GEN}}(q), \vec{\texttt{GEN}}(q^*)))} \tag{11}$$

$$- \frac{1}{1 + \exp(\cos(\vec{\texttt{GEN}}(q), \vec{\texttt{GEN}}(q^-)))}. \tag{12}$$

This pairwise loss function trains the encoder to improve the cosine similarities between positive query pairs (q, q^*) and reduce those between negative pairs (q, q^-).

Picking q^- at random includes many trivial negatives that are easy to distinguish and less informative. GEN Encoder uses noise-contrastive estimation (NCE) to pick adversarial negatives [23]:

$$q^- = \text{argmax}_{q' \in \text{batch}} \cos(\vec{\texttt{GEN}}(q), \vec{\texttt{GEN}}(q')). \tag{13}$$

It picks the negative q^- in the current training batch that is the most similar to q at the current learned version of GEN Encoder.

Multi-Task Learning. The weak supervision signals from co-clicks inevitably includes noise and may be biased towards the search engine's existing ranking system. To learn a generic intent representation, the second learning phase adds two more human labeled paraphrase classification tasks, one on queries and one on questions, in a multi-task learning setting.

The query paraphrase dataset includes web search query pairs and expert labels: $\{q_i, q'_i, y_i\}$; $y_i = +1$ if (q_i, q'_i) share the exact same intent (paraphrase) and $y_i = -1$ otherwise. The question paraphrase dataset is the same except it includes natural language questions $\{qe_i, qe'_i, y_i\}$.

GEN Encoder is trained using logistic regression with cosine similarities as the loss:

$$l_{\text{q-para}} = \sum_i \frac{1}{1 + \exp(y_i \times \cos(\vec{\texttt{GEN}}(q_i), \vec{\texttt{GEN}}(q'_i)))}, \tag{14}$$

$$l_{\text{qe-para}} = \sum_j \frac{1}{1 + \exp(y_j \times \cos(\vec{\texttt{GEN}}(qe_j), \vec{\texttt{GEN}}(qe'_j)))}. \tag{15}$$

The two tasks are combined with co-click weak supervisions using multi-task learning:

$$l_{\text{multi-task}} = l_{\text{co-click}} + l_{\text{q-para}} + l_{\text{qe-para}}. \tag{16}$$

The first phase learns end-to-end from the large scale co-click signals from Bing search logs. The second phase fine-tunes the model with high-quality but expensive human labels.

4 INTRINSIC EVALUATIONS

Before demonstrating the potential of Gen Encoder in downstream applications, our first set of experiments directly evaluate the quality of the learned intent representations using *intrinsic* evaluations.

4.1 Experimental Methodologies

This section first describes the experimental methodologies, including the task, training data, baselines, and implementation details.

Evaluation Tasks. Our experiments follow the common intrinsic evaluations on word embeddings [12, 30, 34] and evaluate the intent representations by their ability to *model query similarities at the user intent level via their distances in the representation space.* Three Intent Similarity modeling datasets are used.

General is the main dataset. It first samples a set of target queries from Bing search log. Then for each target query, several candidate queries appearing frequently with it in sessions are sampled. Human experts label the candidate queries based on their intent similarity with the target query in three scales: "Good", meaning they have the same intent and require the same search results; "Fair", which means the two queries are similar and some documents can satisfy both; "Bad" means the users are looking for different things for the two queries. Thus this is a graded scale of query intent similarity, i.e. a ranking based problem for evaluation.

Easy is an easier dataset. It focuses on trivial word level variations and spelling errors. It is also a ranking dataset.

Hard is the "adversial" dataset. It includes the failure cases from previously developed query representations in our system. It is a classification dataset with "Good" and "Bad" labels on query pairs.

Table 1 lists the details and examples of the three datasets. They are sampled differently and there is little overlap between them.

Evaluation Strategy. For all evaluated representation models, they are first used to represent the queries. Then the distances between the query pairs' representations are evaluated against their similarity labels. No information from Intent Similarity queries or labels is allowed to be used in model training or validation.

The ranking datasets (General and Easy) are evaluated by NDCG with no cut-off. The classification dataset (Hard) uses AUC score. The statistically significant differences between methods are tested using permutation (Fisher's randomization) test with $p < 0.05$.

Training Data. The two phase training of GEN Encoder uses co-click queries, query paraphrases, and question paraphrases.

Co-click Queries are sampled from six month of Bing search logs. Queries that lead to clicks on the same URL are collected to

Table 2: Datasets used to train GEN Encoder in the two-stage setting: first weak-supervision and then multi-task learning.

Stage	Dataset	Training	Validation	Testing	Labels
Weak-Supervision	Co-click Queries	≈200M groups	≈10K groups	–	50%+/50%-
Multi-Task	Co-click Queries	≈400K groups	≈10K groups	–	50%+/50%-
	Query Paraphrases	≈800K pairs	≈10K pairs	≈10K pairs	≈25%+/75%-
	Question Paraphrases	≈350K pairs	≈10K pairs	≈10K pairs	≈30%+/70%-

Table 3: GEN Encoder Parameters.

Parameter	Dimension	Description
$\vec{t}_{emb}$	1M*200	Word Embedding
$w_{g^t_{hy}}$	200*200	Word Highway Gate
$w_{p^t_{hy}}$	200*200	Word highway Projection
$\vec{c}$	1000*200	Character Embedding
CNN	[5,100]	Char-Ngram CNN
$w_{g^C_{hy1}}$, $w_{g^C_{hy2}}$	100*100	Char Highway Gate
$w_{p^C_{hy1}}$, $w_{p^C_{hy2}}$	100*100	Char Highway Projection
Bi-GRU Layer	512	RNN Encoder
w^q_{rs}	1324*100	ResNet on Bi-GRU&BOW

form co-click queries. URLs with more than five unique click queries are filtered out because they may reflect more than one intent. The sample includes about 200 Million co-clicked query groups.

Query Paraphrasing contains about 0.8 million manually labeled query pairs with whether they are paraphrases of each other.

Question Paraphrasing has about 0.4 million question pairs manually labeled with whether they are paraphrases of each other.

Table 2 lists the statistics of the three datasets.

Baselines include various representation methods, including bag-of-words, relevance-based embeddings, representation-based neural ranking models, and other generic text representations.

Bag-of-word baselines include the following.

- TF-IDF BOW is the classic discrete bag-of-words based query representation. Its IDF is calculated from the search log.
- Emb BOW (d) is the embedding-based language model [39]. It uses GloVe embeddings pre-trained on documents [26].
- Emb BOW (q) is the embedding-based language model [39], with embeddings trained on search queries [24].

Recent research suggests that distributed representations trained on pseudo relevance feedback are more effective in search tasks [10, 40]. We use an enhanced version of relevance language model [40] (RLM+), which we trained on relevance feedback (user clicks) which is better than pseudo relevance feedback, and then average the learned embeddings of query words to the query embeddings [39]. It includes the following two versions.

- RLM+ (t) is trained on the query and clicked title pairs.
- RLM+ (u) is trained on the query and clicked URL pairs.

Representation based neural ranking baselines include:

- CDSSM, a representation-based neural ranking model with character n-grams and CNN's [32];
- Seq2Seq, the Bi-GRU encoder-decoder model widely used in sequence to sequence learning.

CDSSM and Seq2Seq were both trained on query-clicked titles and query-clicked URL's, resulting in four neural ranking models.

Generic text representation baselines include the following:

- BERT Encoder is the encoder part of pre-trained BERT. The best performing setup in our testing is selected, which is the average of BERT base version's last layer [9].
- USE: the Universal Sentence Encoder trained on various language understanding tasks by multi-task learning [6].

The publicly released versions of BERT[2] and USE[3] are used as they are, in order to provide a better understanding of the effects of different training signals: surrounding text (BERT), language semantic understanding tasks (USE), and user clicks (GEN Encoder). Fine-tuning or learning deeper architectures using our training data or combining these approaches are reserved for future research.

We implemented all other baselines following best practises suggested in previous research. All methods, if applicable, were trained using the same search log sample, the same scale of training data, and the same hyper-parameter settings. They are then evaluated exactly the same with GEN Encoder.

RLM+ is the most related query representation method designed for information retrieval tasks [40]. It performed better than the locally trained word embeddings [10] and discrete PRF expansions [18], making it the main IR baseline in our evaluations. USE is the most related generic representation baseline and has shown strong performances on language understanding tasks.

Implementation Details The parameters of GEN Encoder and their dimensions are listed in Table 3. Training uses Adam optimizer, 1e-4 learning rate, and 256 batch size. All parameters are learned end-to-end. Learning in each phase is concluded base on validation loss. On a typical GPU, the first phase takes about 300 hours per epoch and converges after 1 epoch; the second phase takes one hour per epoch and several epochs to converge. The multi-task learning phase randomly mixes the training data from three datasets in each mini-batch. On a typical GPU it takes about 15 milliseconds to infer the GEN Encoding for an average query.

4.2 Evaluation Results

Two experiments are conducted to study GEN Encoder's accuracy and source of effectiveness using the intent similarity datasets.

4.2.1 Accuracy on Modeling Intent Similarity. Table 4 shows the overall evaluation results. Baselines are grouped as bag-of-words (BOW), neural matching models (CDSSM, Seq2Seq), relevance-based embeddings (RLM+), and generic representations (BERT, USE). For

[2]https://github.com/google-research/bert
[3]https://tfhub.dev/google/universal-sentence-encoder/2

Table 4: Performances on Query Intent Similarity Modeling. Relative difference over TF-IDF BOW are presented in percentages. †, ‡, §, ¶ marks the statistical significant improvements over TF-IDF BOW†, Emb BOW (q)‡, RLM+ (title)§ and USE¶ (p<0.05).

	Query Intent Similarity						Paraphrase Classification			
Method	**General (NDCG)**		**Easy (NDCG)**		**Hard (AUC)**		**Question (AUC)**		**Query (AUC)**	
TF-IDF BOW	0.4969	–	0.8047	–	0.4740	–	0.6869	–	0.8992	–
Emb BOW (d)	0.4842	−2.54%	0.8567†	+6.47%	0.5059	+6.73%	0.7093†	+3.26%	0.9146	+1.71%
Emb BOW (q)	0.4834	−2.71%	0.8583†§	+6.66%	0.5055	+6.65%	0.7111†	+3.52%	0.9186	+2.15%
CDSSM (t)	0.3830	−22.92%	0.8237†	+2.36%	0.4476	−5.58%	0.5382	−21.64%	0.5973	−33.57%
CDSSM (u)	0.3857	−22.37%	0.8281†	+2.91%	0.4700	−0.84%	0.5067	−26.23%	0.6112	−32.03%
Seq2Seq (t)	0.4606	−7.30%	0.8593†‡§	+6.79%	0.5300	+11.81%	0.7184†	+4.58%	0.8724	−2.98%
Seq2Seq (u)	0.4499	−9.45%	0.8584†‡§	+6.68%	0.5106	+7.71%	0.7044	+2.55%	0.8721	−3.02%
RLM+ (t)	0.4985†‡¶	+0.33%	0.8570†	+6.50%	0.5036	+6.24%	0.7343†‡	+6.91%	0.9562†‡¶	+6.33%
RLM+ (u)	0.4890‡	−1.59%	0.8592†‡§	+6.77%	0.4938	+4.17%	0.7138†	+3.91%	0.9331†‡	+3.77%
BERT Encoder	0.4643	−6.56%	0.8585†‡§	+6.69%	0.4977	+5.00%	0.7352†‡	+7.04%	0.9027	+0.39%
USE	0.4958‡	−0.21%	0.8635†‡§	+7.31%	0.5675†§	+19.72%	0.7974†‡§	+16.09%	0.9477†‡	+5.39%
GEN Encoder	0.5244†‡¶	+5.53%	0.8688†‡§¶	+7.97%	0.6667†‡§¶	+40.64%	0.8486†‡§¶	+23.55%	0.9863†‡§¶	+9.68%

the generic representations, performance on the Paraphrase Classification tasks are presented, but more for reference, as they were not fine-tuned on paraphrase labels. All methods were evaluated on Query Intent Similarity exactly the same.

The embedding methods Emb BOW and the representation-based neural ranking methods (CDSSM and Seq2Seq) do not outperform TF-IDF BOW. Embeddings trained on surrounding texts or representations learned to match documents are not designed to represent search intents. RLM+ performs better than BOW models in the majority of datasets. Embeddings trained using relevance signals are more effective than embeddings trained on surrounding contexts in modeling user intent: RLM+ uses the same average of word embeddings with Emb BOW, but its embeddings are trained on the relevance feedback signals which better reflects information needs [40].

USE shows strong performances across the datasets, while BERT Encoder's effectiveness is more mixed. The two differ in two aspects. First, USE is designed to learn general representations and focuses on building a universal representation space [6]. In comparison, BERT is a sequence to sequence model; much of the model capacity resides in the cross-sequence multi-head attentions, instead of general representation [9]. Second, USE leverages multi-task learning to improve the learned representation's "Universal" generalization ability [6] while BERT focuses on single task.

GEN Encoder outperforms all baselines on all datasets. Its advantages are significant, stable, and stronger on harder datasets. It is the only one that significantly outperforms TF-IDF (BOW) on General, and outperforms all other methods on Hard with large margins (at least 17%). GEN Encoder is more effective on all intent similarity datasets compared to RLM+, the most related IR-style query representation method. It also significantly outperforms USE, the previous state-of-the-art in general representation learning. The next experiment studies the reasons for these improvements.

4.2.2 Source of Effectiveness. This experiment studies the contribution of GEN Encoder's training strategies and neural architecture.

Learning Ablations. The top half of Table 5 shows the model performances with different subsets of training data. It varies the percentage of co-click data used, as shown on the left of the Method column, and the set of paraphrase labels used in the second phase, as shown on the right of the Method column.

The co-click weak supervision is the most important for GEN Encoder. When no co-click is used (0%), there is not sufficient data to learn a meaningful intent representation space. When all co-click signals are used (100%), GEN Encoder reaches its peak performances on General and Easy datasets, even without the second learning phase. The noise-contrastive estimation (NCE) also improves the accuracy with more informative negative training samples.

Multi-task learning is crucial to the model generalization and performances on Hard. It improves the AUC on Hard by about 10%, full model vs. (100%, none). On the other hand, if only one task is used in the second phase, GEN Encoder performs well on the corresponding task: (100%, query) on Query Paraphrase and (100%, question) on Question Paraphrase, but worse on all other tasks than only using the first phase (100%, none).

Encoder Ablations. The lower half of Table 5 shows the performances of different encoder architectures: Avg-Emb is the average of word embeddings; Bi-GRU is the vanilla Bi-GRU on word embeddings; no char-embedding discards the character embedding component; no highway discards all highway layers. All encoder components contribute; discarding any of them often reduces performances, mostly noticeably on Hard and Question Paraphrase.

This experiment demonstrates the importance of learning strategy in building generic intent representations. Trained on the right data, a simple embedding (Avg-Emb) outperforms more complex models trained for other purposes. Avg-Emb even reaches the performance of USE, which uses a deep transformer architecture and multi-task learning, but is not trained for search tasks.

An interesting finding in our ablation study is that the encoder may not have sufficient capacity to consume weak supervision signals at this scale. Its accuracy plateaus with about 10% co-click data (20 million). The ability to learn from user clicks enables large scale weak supervision; to fully leverage the potential of this available scale may require training deep architectures more efficiently, which is a future research direction.

Table 5: Performances of GEN Encoder variations. Relative performances and statistically significant differences* are compared to the full GEN Encoder. The varied training data in the two learning phases are listed in the brackets: (first, second).

Different Learning Strategies, Same Gen Encoder										
	Query Intent Similarity						Paraphrase Classification			
Method	Normal (NDCG)		Easy (NDCG)		Hard (AUC)		Question (AUC)		Query (AUC)	
(1%, none)	0.5136*	−2.05%	0.8711*	+0.26%	0.5781*	−13.28%	0.7646*	−9.90%	0.9273*	−5.98%
(10%, none)	0.5230*	−0.27%	0.8719*	+0.36%	0.6018*	−9.73%	0.7832*	−7.71%	0.9503*	−3.65%
(100%, none)	0.5278*	+0.65%	0.8734*	+0.52%	0.6059*	−9.12%	0.7795*	−8.15%	0.9519*	−3.48%
(100% no NCE, none)	0.5225*	−0.36%	0.8711*	+0.26%	0.5880*	−11.79%	0.7694*	−9.34%	0.9515*	−3.52%
(0%, query)	0.4273*	−18.51%	0.8376*	−3.59%	0.4943*	−25.85%	0.5680*	−33.07%	0.9532*	−3.35%
(0%, question)	0.4232*	−19.30%	0.8457*	−2.66%	0.4373*	−34.40%	0.6465*	−23.82%	0.7612*	−22.81%
(0%, multi-task)	0.5101*	−2.72%	0.8661*	−0.31%	0.5220*	−21.70%	0.7912*	−6.77%	0.9725*	−1.40%
(100%, query)	0.5056*	−3.58%	0.8667*	−0.24%	0.5955*	−10.68%	0.6969*	−17.88%	0.9869	+0.06%
(100%, question)	0.5164*	−1.52%	0.8679*	−0.10%	0.5917*	−11.24%	0.8496	+0.11%	0.9585*	−2.81%
full model	0.5244	–	0.8688	–	0.6667	–	0.8486	–	0.9863	–
Different Encoder Architecture, Same Two-Phase Learning										
	Query Intent Similarity						Paraphrase Classification			
Method	Normal (NDCG)		Easy (NDCG)		Hard (AUC)		Question (AUC)		Query (AUC)	
Avg-Emb	0.5081*	−3.11%	0.8612*	−0.87%	0.5670*	−14.95%	0.8286*	−2.37%	0.9620*	−2.46%
Bi-GRU	0.5055*	−3.59%	0.8633*	−0.63%	0.5494*	−17.59%	0.7665*	−9.68%	0.9681*	−1.84%
no char-embedding	0.5182*	−1.17%	0.8726*	+0.43%	0.6292	−5.61%	0.7797*	−8.13%	0.9546*	−3.21%
no highway	0.5254*	+0.20%	0.8686*	−0.02%	0.6172	−7.43%	0.7459*	−12.11%	0.9495*	−3.72%
full model	0.5244	–	0.8688	–	0.6667	–	0.8486	–	0.9863	–

Table 6: Statistics of ANN at different distance radius (0.15, 0.10, 0.05): percent of queries that have ANN neighbors (Coverage), average number of ANN neighbors (# Neighbor) among covered queries, and the fraction of ANN neighbors sharing same user intents (Co-Intent %).

	Coverage %			# Neighbor			Co-Intent %		
	0.15	0.10	0.05	0.15	0.10	0.05	0.15	0.10	0.05
Head	96.1	90.0	79.8	5.05	4.92	4.32	92	100	100
Torso	88.3	75.2	59.7	3.77	3.57	3.17	80	90	99
Tail	57.9	32.9	15.9	2.91	2.74	2.32	47	59	80

5 ALLEVIATING THE TAIL SPARSITY

Much of a search engine's intelligence comes from its users. User click behavior from previous observations of a query is one of the most effective relevance signals [1]; previous clicked queries on a document is among the most informative document representation in relevance ranking [41]. However, the long tail nature of search traffic means many queries are rarely, if ever, observed,

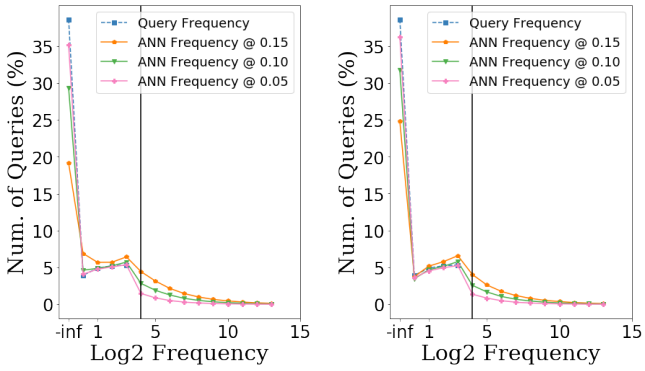
This section demonstrates how GEN Encoder alleviates the long tail challenge by mapping a rare query to previously observed queries sharing the same intent by using approximate nearest neighbor (ANN) search. We first describe the setup of our experiment, followed by quantitative and qualitative analyses of the tail alleviation, and then discuss how this phenomenon improves the coverage of our online question answering capabilities.

ANN Index and Queries. We built an ANN index of the GEN encodings of 700 million queries sampled from six months, using the HNSW algorithm [20]. In a typical parallel computing environment (feasible in academic settings), the index performs an ANN lookup within at most 10 milliseconds. For this study, we randomly sampled one million queries occurring on one day some time after the ANN index is built, with navigational and adult queries filtered out. About 5% of them are head (appear more than 2^{15} times), 40% are torso (2^5 to 2^{15}), and 55% are tail queries ($\leq 2^4$). We retrieved the top ten nearest neighbors for each query within varying cosine distance radius (0.15, 0.10, or 0.05).

ANN Coverage and Accuracy. Table 6 shows the statistics of the ANN search results for the one million queries. The Co-Intent % evaluates the fraction of ANN queries sharing the exact same intent as the search query. We asked human judges to evaluate the retrieved nearest neighbors of 100 randomly sampled queries, comprised of 3 head, 47 torso, and 50 tail queries. Three experts labeled them with an average of 0.717 Cohen's Kappas (high agreement). Disagreements were resolved by majority.

The ANN search covers most head and torso queries. Table Table 6 shows the coverage on tail queries is significantly more sparse, as expected, but still half of them have ANN queries within radius of 0.15, with average $2-3$ ANN per query. However, not all of the nearest neighbors are co-intent queries (about 47% are in the tail), so each of the 57.9% of tail queries which have neighbors will have about 1.37 ($2.91 \times 47\%$) co-intent neighbors. This result confirms that many tail queries are rare ways of expressing a search intent that has been expressed by another query before [11], and that ANN search in the GEN encoder space (ANN-GEN) can efficiently retrieve such co-intent queries for a large fraction of tail queries.

Alleviating the Tail. A direct impact of ANN-GEN is to alleviate the sparsity of tail queries by combining observations from the semantically similar queries found using ANN search. For example,



(a) All Frequency **(b) Penalized Frequency**

Figure 2: The distributions of tail query frequency and their ANN frequency at different neighbor radius. Queries and ANNs are binned by their frequencies in the six-month period before their date. X-axis marks the $\log_2$ frequency; -inf indicates unseen queries. ANN Frequency in (b) is penalized by co-intent accuracy (e.g. 0.47) from Table 6. Y-axis is fraction of corresponding bin in total traffic. The tail range ($\leq 2^4$) are marked by vertical lines.

the user click behaviors from approximate neighbors of a tail query can be used to improve its relevance ranking.

Figure 2 illustrates how bringing in approximate neighbors of a search query can make the tail less sparse. The Query Frequency counts the occurrences of tail queries in a very large sample from the span of six months right before the day the query was sampled from. The ANN Frequency is the total number of occurrences if the query's approximate neighbors are introduced and their observations are added to the search query's frequency. Since we filtered out navigational traffic, the line with square marks in Figure 2 shows that about 39% of our total query sample never appeared in the previous six month window, and thus there is no feedback signals for them at all. ANN alleviated the sparsity on extreme rare queries significantly: the ≈40% of queries which are unseen in the log drops by half at the 0.15 radius threshold (Figure 2a), or by ≈35% if we account for neighbors which are non-co-intent (Figure 2b).

Influences in Online Production. This ANN search with `GEN Encoder` has been deployed in various components of Bing search systems. One of its many applications is to improve the coverage of the search engine's online question answering by aligning unanswerable questions to their answerable semantically equivalent `ANN-GEN` questions. The coverage of an important fraction of online QA pipeline was *doubled* without loss of answer quality.

6 UNDERSTANDING SEARCH BEHAVIOR

Another advantage of distributed representations is that the distance in the embedding space may reflect certain meaningful relationships between the points. This section analyzes the distances in `Gen Encoder`'s representation space, and then shows the distances between query pairs reflect interesting behaviors in user sessions.

Data. The dataset used is one million search sessions sampled from one day's search logs. Sessions are delineated by a 30-minute gap. Standard non-navigational and work-safe filters are applied. Sessions with less than three queries or no click are filtered.

Table 7: Spearman's rank-order correlations between representation distances and expert labels in the four intent-transit classes: "Topic Change" (0), "Explore" (1), "Specify" (2), and "Paraphrase" (3). Related includes (1-3) and Middle is (2-3). `Human` is the average performances of authors' labels.

Method	All (100)	Related (55)	Middle (45)
`TF-IDF BOW`	.642	.459	.285
`Emb BOW (q)`	.522	.351	.123
`BERT Encoder`	.626	.436	.327
`RLM+ (t)`	.705	.425	.242
`USE`	.678	.383	.123
`GEN Encoder`	.800	.580	.429
`Human`	.859	.776	.620

Distances Distributions. Figure 3 plots the distributions of distances of query pairs in `Emb BOW (q)`, `BERT Encoder`, and `GEN Encoder`. As expected, query pairs appearing together in sessions are more similar. However, the distributions of distances in these three methods differ considerably: The adjacent query pairs in sessions (query reformulations) follow a strong and flat bi-modal distribution in `GEN Encoder`, very different from the distributions from previous embedding methods.

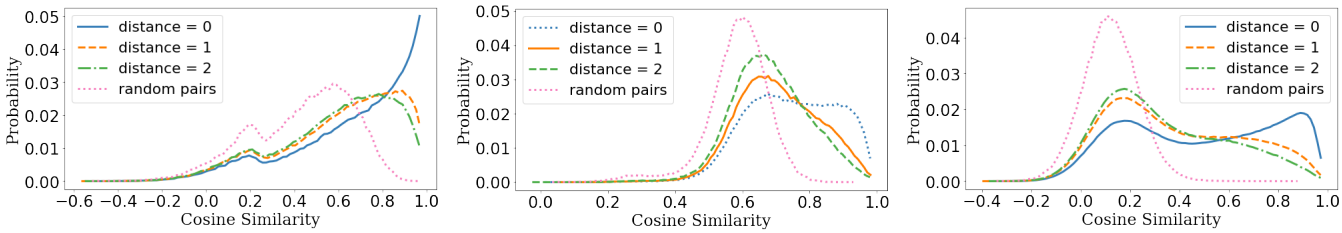
Human Analyses. To understand this bi-modal distribution, we conduct human analyses on the relations between these adjacent queries. One hundred adjacent query pairs are randomly sampled from these sessions. Three judges label them into four categorizes of information seeking behaviors, following previous research in session understanding [5, 14]

(1) `Topic Change`: The two queries are unrelated.
(2) `Explore`: The second query explores within topic space of the previous query.
(3) `Specify`: The second query narrows down the intent of the previous query.
(4) `Paraphrase`: The second query has the exact same intent of the previous query.

They agree at 0.64 Cohen's Kappa, which is a reasonable high agreement when labeling four classes. Disagreements were resolved by majority. Ties are broke by a fourth judge.

Table 7 lists the correlations between representation similarities and human labels. Table 8 lists some example query pairs. `GEN Encoder` correlates the most with human labels. As shown in the Related and Middle columns, Though all methods seem able to differentiate "Topic Change" from the rest, `GEN Encoder` does the best at identifying "Explore" and "Specify" categories, which capture interesting information seeking behaviors in search sessions.

This result further demonstrates the quality of the `GEN Encoder`'s generic intent representation space and suggests its future applications in analyzing and understanding user's information seeking behavior.



(a) Emb BOW (q) **(b) BERT Encoder** **(c) GEN Encoder**

Figure 3: The distributions of cosine similarities between queries in search sessions that are adjacent (1), separated by one (2) or two (3) other queries in the session, or between random sampled queries. The distance is the number of queries the user issued in between the two queries. Query pairs were randomly sampled in Bing search sessions with more than two queries.

Table 8: Examples of Topic Change, Exploratory and Specification query reformulation pairs, labeled by our experts. Their cosine similarities in various distributed representation space are listed. Emb is Emb BOW (q) and RLM+ is on title (t).

First Query	Second Query	Label	Emb	BERT	RLM+	GEN
"donald norman design of everyday things"	"size an image in html"	Topic Change	0.80	0.61	0.62	0.26
"facial lotion containing menthol and phenol"	"sarna lotion"	Explore	0.65	0.67	0.51	0.61
"fitness social fresno"	"orange theory fitness fresno"	Explore	0.90	0.80	0.84	0.81
"how to change autofill settings"	"how to change autofill settings chrome"	Specify	0.99	0.92	0.96	0.84
"cdma"	"cdma in a cell phone"	Specify	0.53	0.76	0.50	0.75

7 CONCLUSION AND FUTURE WORK

This paper presented GEN Encoder, a neural system that learns a distributed representation space that captures user intent. Search queries are often insufficient to express what a user wants, but user clicks provide implicit feedback signals for their search intent. Instead of manually defining intent categories or an ontology, GEN Encoder utilizes user clicks as weak supervision of user intent, and learns end-to-end its encoder architecture by mapping co-click queries together in its representation space. We further fine-tuned GEN Encoder to improve it's generality.

We used an intrinsic evaluation task – query intent similarity modeling – to evaluate the quality of query representations. GEN Encoder showed significant, robust, and large margin improvements over a wide range of previous text representation methods, and its advantages thrive on the hard dataset. A series of ablation studies illustrate: 1) the necessity of user feedback signals in learning a generic intent representation space, 2) the differences between representations learned from surrounding texts and representations learned from user clicks, and 3) the improved generalization ability from multi-task learning.

An effective distributed representation of user intent has many implications in information retrieval. This paper demonstrates how it helps alleviate the sparsity of long tail queries with approximate nearest neighbor search in the continuous space. Though a query may never appear before, its search intent might have been expressed by other queries in the search log. We show that, with an index of 700 million queries' GEN encodings, ANN search finds such co-intent queries with high coverage, sufficient accuracy, and practical latency. Incorporating the observations from ANN queries significantly alleviates the sparsity of tail traffic; it reduces the fraction of unseen queries by 35%-50% relatively. In practice, efficient ANN search with effective generic intent embeddings has wide impact in various components of Bing search systems.

The last experiment demonstrates the emergent behavior of GEN Encoder's representation space, whose distances separate commonly recognized user behaviors in search sessions. The distances of query reformulations in search sessions follow a strong bi-modal distribution, and correlate well with the four types of query reformulation behaviors: "Topic Change", "Exploratory", "Specification", and "Paraphrase". This suggests GEN Encoder can be used to understand user's information seeking behaviors and provide more immersive search experiences. For example, it can be used to identify sessions that will follow a trajectory such as learning a new topic or completing a specific task; which opens the opportunity for more direct answers, more engaging question suggestions, and a more conversational interaction with the search engine.

Having a meaningful generic representation for user search intents suggests many downstream applications; this paper can only demonstrate a few of them. We are delighted to make Gen Encoder available through https://aka.ms/GenEncoder to facilitate more exploration of its potential usage.

8 ACKNOWLEDGEMENT

We want to thank Tong Wang, Subhojit Som, Maria Kang, Kamara Benjamin and Marc Rapaport for helping with ANN search experiments and human evaluation. We also thank Guoqing Zheng and Susan Dumais for providing valuable feedback for this paper.

REFERENCES

[1] Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 19–26.

[2] Michael Bendersky, Donald Metzler, and W. Bruce Croft. 2011. Parameterized concept weighting in verbose queries. In *Proceedings of the 34th annual international ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR 2011)*. ACM, 605–614.

[3] Andrei Broder. 2002. A taxonomy of web search. In *ACM Sigir forum*, Vol. 36. ACM, 3–10.

[4] Andrei Z Broder, Marcus Fontoura, Evgeniy Gabrilovich, Amruta Joshi, Vanja Josifovski, and Tong Zhang. 2007. Robust classification of rare queries using web Knowledge. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007)*. ACM, 231–238.

[5] Ben Carterette, Evangelos Kanoulas, Mark Hall, and Paul Clough. 2014. Overview of the TREC 2014 session track. In *Proceedings of The 23rd Text Retrieval Conference (TREC 2014)*.

[6] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175* (2018).

[7] W Bruce Croft, Donald Metzler, and Trevor Strohman. 2010. *Search Engines: Information Retrieval in Practice*. Addison-Wesley Reading.

[8] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM 2018)*. ACM, 126–134.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[10] Fernando Diaz, Bhaskar Mitra, and Nick Craswell. 2016. Query Expansion with Locally-Trained Word Embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*. 367–377.

[11] Doug Downey, Susan Dumais, and Eric Horvitz. 2007. Heads and tails: studies of web search with common and rare queries. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 847–848.

[12] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: The concept revisited. *ACM Transactions on information systems* 20, 1 (2002), 116–131.

[13] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 55–64.

[14] Ahmed Hassan, Ryen W White, Susan T Dumais, and Yi-Min Wang. 2014. Struggling or exploring?: disambiguating long search sessions. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 53–62.

[15] Jian Hu, Gang Wang, Fred Lochovsky, Jian-tao Sun, and Zheng Chen. 2009. Understanding user's query intent with wikipedia. In *Proceedings of the 18th international conference on World wide web*. ACM, 471–480.

[16] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management (CIKM 2013)*. ACM, 2333–2338.

[17] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-Aware Neural Language Models.. In *AAAI*. 2741–2749.

[18] Victor Lavrenko and W Bruce Croft. 2001. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*. ACM, 120–127.

[19] Xiao Li, Ye-Yi Wang, and Alex Acero. 2008. Learning query intent from regularized click graphs. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 339–346.

[20] Yury A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence* (2018).

[21] Donald Metzler and W Bruce Croft. 2005. A Markov random field model for term dependencies. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2005)*. ACM, 472–479.

[22] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 27th Advances in Neural Information Processing Systems 2013 (NIPS 2013)*. NIPS, 3111–3119.

[23] Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in neural information processing systems*. 2265–2273.

[24] Eric Nalisnick, Bhaskar Mitra, Nick Craswell, and Rich Caruana. 2016. Improving document ranking with dual word embeddings. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 83–84.

[25] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Jingfang Xu, and Xueqi Cheng. 2017. Deeprank: A new deep architecture for relevance ranking in information retrieval. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 257–266.

[26] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP 2014)*. 1532–1543.

[27] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

[28] Navid Rekabsaz, Mihai Lupu, Allan Hanbury, and Hamed Zamani. 2017. Word Embedding Causes Topic Shifting; Exploit Global Context!. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1105–1108.

[29] Gerard Salton and Chris Buckley. 1990. Improving retrieval performance by relevance feedback. *Journal of the American society for information science* 41, 4 (1990), 288–297.

[30] Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 298–307.

[31] Dou Shen, Jian-Tao Sun, Qiang Yang, and Zheng Chen. 2006. Building bridges for web query classification. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 131–138.

[32] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 373–374.

[33] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Advances in neural information processing systems (NeuIPS 2015)*. 2377–2385.

[34] Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. *arXiv preprint arXiv:1804.07461* (2018).

[35] Chenyan Xiong and Jamie Callan. 2015. Query expansion with Freebase. In *Proceedings of the fifth ACM International Conference on the Theory of Information Retrieval (ICTIR 2015)*. ACM, 111–120.

[36] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th annual international ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR 2017)*. ACM, 55–64.

[37] Xiaoxin Yin and Sarthak Shah. 2010. Building taxonomy of web search intents for name entity queries. In *Proceedings of the 19th international conference on World wide web (WWW 2010)*. ACM, 1001–1010.

[38] Hamed Zamani and W Bruce Croft. 2016. Embedding-based query language models. In *Proceedings of the 2016 ACM international conference on the theory of information retrieval (ICTIR 2016)*. ACM, 147–156.

[39] Hamed Zamani and W Bruce Croft. 2016. Estimating embedding vectors for queries. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval (ICTIR 2016)*. ACM, 123–132.

[40] Hamed Zamani and W Bruce Croft. 2017. Relevance-based word embedding. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2017)*. ACM, 505–514.

[41] Hamed Zamani, Bhaskar Mitra, Xia Song, Nick Craswell, and Saurabh Tiwary. 2018. Neural Ranking Models with Multiple Document Fields. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM 2018)*. 700–708.

[42] Guoqing Zheng and James P. Callan. 2015. Learning to reweight terms with distributed representations. In *Proceedings of the 38th annual international ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR 2015)*. ACM, 575–584.