Recommending Search Queries in Documents Using Inter N-Gram Similarities

Eilon Sheetrit*
seilon@amazon.com
Amazon

Yaroslav Fyodorov
yfyodorov@verizonmedia.com
Yahoo Research

Fiana Raiber
fiana@verizonmedia.com
Yahoo Research

Oren Kurland
kurland@ie.technion.ac.il
Technion

ABSTRACT

Reading a document can often trigger a need for additional information. For example, a reader of a news article might be interested in information about the persons and events mentioned in the article. Accordingly, there is a line of work on recommending search-engine queries given a document read by a user. Often, the recommended queries are selected from a query log independently of each other, and are presented to the user without any context. We address a novel query recommendation task where the recommended queries must be n-grams (sequences of consecutive terms) in the document. Furthermore, inspired by work on using inter-document similarities for document retrieval, we explore the merits of using inter n-gram similarities for query recommendation. Specifically, we use a supervised approach to learn an inter n-gram similarity measure where the goal is that n-grams that are likely to serve as queries will be deemed more similar to each other than to other n-grams. We use the similarity measure in a wide variety of query recommendation approaches which we devise as adaptations of ad hoc document retrieval techniques. Empirical evaluation performed using data gathered from Yahoo!'s search engine logs attests to the effectiveness of the resultant recommendation methods.

CCS CONCEPTS

• **Information systems → Information retrieval**;

KEYWORDS

query recommendation; inter n-gram similarity

ACM Reference Format:
Eilon Sheetrit, Yaroslav Fyodorov, Fiana Raiber, and Oren Kurland. 2021. Recommending Search Queries in Documents Using Inter N-Gram Similarities. In *Proceedings of the 2021 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '21), July 11, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3471158.3472252

*Part of this work was done during Eilon Sheetrit's internship at Yahoo Research.

ICTIR '21, July 11, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8611-1/21/07…$15.00
https://doi.org/10.1145/3471158.3472252

1 INTRODUCTION

Reading a document may incite one's curiosity and trigger some information needs. In the Web setting, seeking information that satisfies these needs is often done by posting queries to a search engine. Accordingly, there is a line of work on predicting which queries a user reading a document might post [5, 9, 19, 24]. This prediction is the basis for *recommending queries* to users reading documents[1]. The recommended queries are in the vast majority of cases those from a query log [5, 9, 19].

Here, we address a novel challenge: recommending n-grams in a document a user is reading as search engine queries. The n-grams are sequences of n consecutive terms in a document. The motivation for engaging in this task is three-fold. First, studies have shown that a query submitted by users after reading a document is likely to appear partially or fully in the document itself [19]. This finding implies to the potential effectiveness of the document being read as a source for recommended queries. The second motivation is the convenience of user interfaces as that in Figure 1: the recommendations are simply highlighted n-grams in the document; clicking on these n-grams launches them as queries in a search engine. The third motivation is the level of explainability provided by the interface. The user can use the context surrounding the recommended n-gram to decide whether it represents well her information need.

To address the novel query-recommendation challenge described above, we present a novel approach that is inspired by work on utilizing inter-document similarities for ad hoc document retrieval (e.g., [21, 23, 28]). Instead of treating potential queries (n-grams) independently of each other, as in past work, when estimating whether they will be used by a user, we study the merits of utilizing their relations. Essentially, our approach enriches the representation of an n-gram by utilizing information induced from associated n-grams in the same document. Since n-grams are short segments of text, such enrichment can be of much merit as we show.

We devise a supervised inter n-gram similarity measure which induces a similarity space where n-grams in the same document that are used as queries are closer to each other than to those in the document that are not used as queries. This is a novel manifestation of the *cluster hypothesis* originally proposed for document retrieval [17][2], which we apply to the query recommendation task.

[1]We use the term "query recommendation" rather than "query suggestion" to differentiate the task we focus on from that of suggesting queries to users of search engines who have already started their search session [2, 4, 12, 46].
[2]The cluster hypothesis states that "closely associated documents tend to be relevant to the same requests" [17].

Figure 1: Example of highlighted n-grams that are recommended queries. A click on an n-gram launches it as a query in a search engine.

Accordingly, we devise a cluster hypothesis test for the query recommendation task which allows to quantify the extent to which co-used recommended queries are similar to each other.

We learn the n-gram similarity measure by framing the similarity-induction task as a learning-to-rank problem [27]. To recommend queries, we rank the n-grams in a document by using our n-gram similarity measure in various ranking methods adapted from work on using inter-document similarities for ad hoc document retrieval.

Empirical evaluation performed using data gathered from Yahoo!'s search engine logs attests to the effectiveness of using inter n-gram similarities, via our proposed similarity measure, to recommend n-grams in documents as queries.

Summary of Contributions. (1) We address a novel task: recommending n-grams *in* a document as search-engine queries for a user reading the document; (2) we present the first (to the best of our knowledge) suite of document-based query-recommendation approaches[3] which utilize inter-query relations for estimating the likelihood that a given recommended query will be used; (3) we devise an inter n-gram similarity measure and use it in a variety of methods for ranking n-grams as recommended queries; (4) we present an in-depth analysis of the merits of the proposed n-gram similarity measure and those of the query recommendation methods that utilize it; and (5) we present a novel type of analysis of document-based recommended queries which focuses on the similarities of (i) search results pages (SERPs) retrieved for them and (ii) click rates on these SERPs.

2 RELATED WORK

In work on recommending search engine queries to a user reading a document, the queries were either extracted from a query log [5, 7, 9, 19] or were generated based on the document text [24]. Using information induced from a query log [5, 9, 19, 24] and the document itself [5, 19, 24] can help in recommending queries. The task of recommending queries from a query log is conceptually similar to the task of entity linking [44]; i.e., linking an n-gram in a text to an entity repository is similar to linking an n-gram in the text to a query in a query log. There is work on using anchor text as a substitute for a query log [11]. In contrast, we recommend n-grams in the document to be used as queries. We rely only on information in the document and some general corpus statistics. Our approach outperforms a supervised method utilizing the document-based features used for recommending queries from a query log [19].

Our approach is based on inter n-gram relations, where n-grams are the candidate queries to recommend. Inter-query relations were used to diversify an initial list of recommended queries which was not created by using inter-query relations [9]. Information about whether queries belong to the same search session was also used for query recommendation [5]. In contrast to our approach, an inter-query similarity measure was not learned nor used to estimate the basic likelihood that a query will be used by the user. Using our similarity measure for diversification is a future direction.

In some work, queries were generated from a given excerpt of text while putting emphasis on the potential retrieval effectiveness of using these queries [24]. This is in contrast to other approaches [5, 9, 19] that are focused, as ours, on predicting which queries a user will use. Some of the features we use for learning an inter n-gram similarity measure are based on predicting query effectiveness.

There is a big body of work on suggesting queries in a search session regardless of a document. Usually, the suggestions are based on a prefix of a query the user typed, and are produced using query log information, where the goal is to infer the current user's search intent (e.g., [2, 4, 12, 46]). In addition, there is work on using co-occurrences of terms in the corpus to suggest queries for search domains without query logs [3]. This approach is conceptually reminiscent of our approach of using inter n-gram similarities in the absence of query logs. However, we recommend n-grams in documents to be used as queries; i.e., the user does not need to formulate her information need while this is not the case in query suggestions. In addition, we use many more features from different sources of information, some of which are based on corpus statistics.

Recently, deep neural networks (specifically, transformers [48]) were used to generate queries from passages so as to use them to expand the passage representation [34, 35]. In contrast to the task we pursue here, the generated queries are not necessarily n-grams in the passage and they are not evaluated based on whether users reading the passage use them as search queries. Furthermore, inter-query relations were not utilized in contrast to our approach.

The task of key phrase extraction is to identify the terms in the document that best describe its content [32]. There is also work on detecting key phrases for question generation [45]. However, we recommend n-grams in documents to be used as queries.

There is work on predicting whether a query submitted by a user to a search engine is relevant to a previously browsed news article [37]. Our task is different: predicting which n-grams in a document (e.g., article) are likely to be used as search queries.

The merits of using the learned similarity function in our methods are demonstrated by contrasting it with other similarity functions in terms of the resultant query-recommendation effectiveness. We also adopt a *cluster hypothesis* test, originally proposed for evaluating inter-document associations [49], to evaluate our inter-n-gram similarity measure. Similar variants of this test were used to evaluate inter-passage [43] and inter-entity [41] similarities.

3 QUERY-RECOMMENDATION FRAMEWORK

Given a document d, we set out to produce a ranked list of n-grams (sequences of consecutive terms); the top m will be recommended as search-engine queries; m is a free parameter.

To effectively rank n-grams which are very short, a rich n-gram representation is called for. Representing a document using information induced from similar documents was shown to be of much

[3]Herein, "document-based query recommendation" refers to the task of recommending queries to a user reading (or having read) a document.

merit in work on ad hoc document retrieval (e.g., [21, 23, 28, 38]). Motivated by this line of work, we study a suite of n-gram ranking approaches that utilize inter n-gram relations.

Considering all possible n-grams in a document as candidate queries may result in a very large pool, especially for long documents and when using n-grams of varying lengths. Thus, we start by creating a pool composed of a subset of n-grams from all those in the document. Then, we rank the n-grams in the pool and recommend the top-m as queries. The ranking methods utilize an inter n-gram similarity measure.

3.1 Selecting Candidates

We create a pool composed of a subset of n-grams from all those that appear in the document. These n-grams are candidate queries for recommendation. We consider as candidate n-grams the longest sequences of entities (of type person, location or organization) [19], proper nouns [24] or noun phrases [24]. In addition, for n-grams which are entities and proper nouns, we also consider all their subsequences. For example, for the entity "Duchess Kate Middleton", we consider "Duchess", "Kate", "Middleton", "Duchess Kate", "Kate Middleton" and "Duchess Kate Middleton" as candidates. Our methods are not committed to specific named entity recognition and part-of-speech tagging methods. (See Section 4.1 for details about the methods used in our experiments.) We note that an n-gram can appear several times in a text with different markups as the markups depend on the context of the n-gram in the text; e.g., the n-gram "JFK" can be marked as an entity of type person (the US president) or of type location (the JFK airport). Yet, we consider the n-gram once and keep the information of all of its occurrences and markups in the document's text. (A thorough analysis about the type of queries issued by users and their occurrences in the documents can be found in Section 4.2.1.)

Henceforth, $\mathcal{G}_d$ denotes the pool of candidates created for document d; g denotes an n-gram. In Section 4.2 we show that this pool creation approach substantially reduces the number of candidate n-grams. Still, the pool includes the vast majority of queries issued by users reading the documents in our dataset.

3.2 Learning Inter N-Gram Similarities

The methods we consider below for ranking the n-grams in the pool described in Section 3.1 utilize an inter-n-gram similarity measure. We now turn to describe how this similarity measure is learned.

The *cluster hypothesis* for ad hoc document retrieval [47] states that closely associated (i.e., similar) documents tend to be relevant to the same requests. A similar hypothesis was stated for entity retrieval [41]. Along these lines, we state a novel cluster hypothesis for ranking n-grams as potential queries:

Cluster Hypothesis for Query Recommendation:
Two similar n-grams in a document are *both* likely to be used, or not, by users reading the document as queries in a search engine.

Standard surface-level similarity measures, as those used for documents and entities, are not likely to capture the desired type of similarity (association) between n-grams. Therefore, we set out to learn an inter-n-gram similarity measure so as to "adhere" to the hypothesis. Raiber et al. [40] applied a similar approach for document retrieval; i.e., an inter-document similarity measure was learned. However, most of their features are not applicable in our setting where similarity is measured between short n-grams.

Table 1: Summary of features used to learn an inter n-gram similarity measure.

Family	Features
Textual Similarities	TFIDF, SentTFIDF
Semantic Similarities	W2V1, W2V2, ESA
Proximity Features	SharedSent, Proximity
N-gram Priors	MaxIDF, AvgIDF, MaxSCQ, AvgSCQ, InTitle, IPos, NgramTFIDF
N-gram Independent Features	DocLen, Entropy

Some of the n-gram ranking approaches we consider utilize clusters of n-grams that are induced using the learned similarity measure. We first present in Section 3.2.1 the clustering technique employed by these approaches, which also guides the learning of the similarity measure. Then, in Section 3.2.2 we discuss the details of learning the inter n-gram similarity measure.

3.2.1 Clustering N-Grams. We use nearest-neighbor clustering as it showed much merit in comparison to other clustering techniques in work on cluster-based document retrieval [21, 38]. A cluster c_g is created for each n-gram, g, in the pool of candidate n-grams, $\mathcal{G}_d$, created from document d. Let $\mathcal{L}_{g;sim}(\mathcal{G}_d)$ be a ranked list of all the n-grams $g' \in \mathcal{G}_d$ induced using $sim(g, g')$; $sim(\cdot, \cdot)$ is the inter n-gram similarity measure that we discuss below. The cluster c_g contains g and the $k-1$ highest ranked n-grams $g' \neq g$ in $\mathcal{L}_{g;sim}(\mathcal{G}_d)$; i.e., each cluster contains k n-grams; $C(\mathcal{G}_d)$ is the resultant set of clusters.

3.2.2 Learning a Similarity Measure. Following the clustering approach just described, we learn and apply an inter n-gram similarity measure as follows. Given a fixed n-gram g ($\in \mathcal{G}_d$), we would like to estimate the similarity $sim(g, g')$ so as to rank n-grams $g' \in \mathcal{G}_d$ with respect to g. If g is (not) likely to be used as a query, we opt for the highly ranked n-grams to be those which are (not) likely to be used as queries. To learn and apply a similarity measure, we represent each pair of n-grams g and g' using a feature vector. We consider 16 features that can be divided into 5 feature families, as summarized in Table 1. We next discuss these features.

Textual Similarities. We use two textual similarity measures. **TFIDF** is the cosine between the TF-IDF vectors representing g and g'. Similarly, **SentTFIDF** is the cosine similarity between the TF-IDF vectors that represent the texts that result from concatenating the sentences in the document d that contain g and g'. The order of concatenation has no effect since we use a bag-of-terms model.

Semantic Similarities. Textual similarity measures may fail to quantify the similarity between short n-grams due to possible vocabulary mismatch. Hence, we use semantic similarity measures. Let $v(w)$ denote the Word2Vec [33] representation of a term w and $|g|$ the number of terms in an n-gram g. We measure the similarity between g and g' using **W2V1** [36]: $\frac{1}{|g|}\sum_{w \in g} \max_{w' \in g'} \cos(v(w), v(w'))$. Since W2V1 is asymmetric, we also use **W2V2**, which is computed by simply switching between g and g' in W2V1. The next semantic similarity measure that we consider is adopted from work on

passage and document retrieval [42]. The **ESA** (Explicit Semantic Analysis [14]) similarity is used by representing each n-gram using a vector that is defined over Wikipedia concepts and computing the cosine similarity between the vectors. Additional details are provided in Section 4.1.

Proximity Features. We assume that n-grams appearing many times next to each other are more likely to be semantically related. Therefore, the features we consider next quantify the proximity between the different occurrences of the n-grams in the document d. **SharedSent** is the number of sentences in d containing both g and g'. **Proximity** is the average proximity between all the occurrences of g and g' [8]. Specifically, let $occ(g, d)$ denote the set of all occurrences of g in d. Proximity is defined as:
$\frac{1}{|occ(g,d)|} \sum_{o_g \in occ(g,d)} \sum_{o_{g'} \in occ(g',d)} \exp\left(-\frac{dist(o_g, o_{g'})^2}{2\sigma^2}\right)$;
$dist(o_g, o_{g'})$ is the minimal distance between the positions of any two terms of the two n-gram occurrences o_g and $o_{g'}$ and σ is a free parameter.

N-gram Priors. The following features quantify the likelihood of g' to be used as a query (in d) independently of g. The first group of n-gram priors are previously proposed effective pre-retrieval query-performance predictors [15]. Originally, these predictors were designed to predict ad hoc document retrieval effectiveness based on information induced from the query and the document corpus without using relevance judgments. They were also used to improve retrieval effectiveness [26, 30]. The four prediction values are computed for g' using information induced from terms in g' with respect to a collection of documents. Note that these predictors are independent of the fixed n-gram g; hence, they serve as priors for g' being used as a query given the collection. **MaxIDF** and **AvgIDF** are the maximum and average of the IDF values of the terms in g' [51]. **MaxSCQ** and **AvgSCQ** are the maximum and average of the TF-IDF values of the terms in g' computed with respect to a collection of documents [51]. Details about the document collection used for experiments are provided in Section 4.1.

We show in Section 4.2 that n-grams appearing at the beginning of the document (a news article in our setting) or in the document's title are more likely to be used as queries by users. Therefore, we define **InTitle**: a binary feature indicating if g' appears in d's title and **IPos**: the inverse position of the first occurrence of g' in d. An additional feature is **NgramTFIDF**: the TF-IDF value of g' in d computed for the n-gram as a whole and not at the term level.

N-gram Independent Features. The features we define next quantify different properties of the document d independently of the pair of n-grams g and g'. Note that these features have an effect only when a non-linear model is used to learn the similarity between n-grams. We use **DocLen**: the number of terms in d, and **Entropy**: the entropy of the term distribution in d computed based on the unsmoothed unigram language model induced from d; Entropy quantifies content repetition in the document.

3.3 Ranking N-Grams

We next describe methods of ranking the n-grams in the pool, $\mathcal{G}_d$. The higher an n-gram is ranked, the more likely it is presumed to be used as a query. We study six n-gram ranking methods which we adapted from work on ranking documents in response to a query. The methods are divided into two groups: those that use the learned similarity measure from Section 3.2 to cluster n-grams and those that use the measure for ranking n-grams without utilizing clusters.

Cluster-Based Approaches. Two classes of cluster-based document retrieval methods were proposed in the literature. The methods in the first class rank clusters based on their relevance to the query, and then transform the ranking of clusters into a ranking of documents by replacing each cluster with its constituent documents while omitting repeats [22, 29, 38]. The methods in the second class use information induced from clusters to enrich the representation of documents [21, 25, 28]. We consider several n-gram ranking methods that are direct adaptations of document ranking techniques from these two classes.

Let $\mathcal{L}_{f_{init}}(\mathcal{G}_d)$ be an initially ranked list of all n-grams $g \in \mathcal{G}_d$ which was produced using *some* n-gram ranking function f_{init} based on the presumed likelihood of the n-grams to be used as queries. For example, the n-grams can be ranked based on the position of their first occurrence in d or by using a learning-to-rank approach that integrates various features. Details about the ranking functions f_{init} we use for experiments are provided in Section 4.1. We then use a rank-to-score transformation [10] to initially score g:

$$S_{Init}(g) \stackrel{def}{=} \frac{1}{rank(g; \mathcal{L}_{f_{init}}(\mathcal{G}_d)) + \nu}; \quad (1)$$

$rank(g; \mathcal{L})$ is g's rank in the list $\mathcal{L}$ and ν is a free parameter; the rank of the highest ranked n-gram is 1. Some of the n-gram ranking methods utilize $C(\mathcal{G}_d)$: the set of n-gram clusters created from the pool of n-grams, $\mathcal{G}_d$, using the approach described in Section 3.2.

The **GM** [29] and **AM** methods, which belong to the first class of cluster-based retrieval methods, score each cluster $c \in C(\mathcal{G}_d)$ using $S_{GM}(c) \stackrel{def}{=} \prod_{g \in c} S_{Init}(g)^{\frac{1}{k}}$ and $S_{AM}(c) \stackrel{def}{=} \frac{1}{k} \sum_{g \in c} S_{Init}(g)$: the geometric and arithmetic mean of the initial retrieval scores of its constituent n-grams, respectively.

Another approach in this class, **CLTR**, is inspired by a state-of-the-art learning-to-rank approach to ranking document clusters [38]. Specifically, we represent each pair of a document and an n-gram cluster as a feature vector[4]. Since the features proposed by Raiber and Kurland [38] are not applicable to short n-grams, we use a different set of features described in Section 4.1. The cluster-ranking model can be trained using any learning-to-rank approach.

The **Interf** [21] method belongs to the second class of cluster-based methods. An n-gram g that is not highly ranked in $\mathcal{L}_{f_{init}}(\mathcal{G}_d)$, but is highly similar to n-grams that are initially highly ranked, should also be highly ranked according to our cluster hypothesis. Therefore, Interf incorporates an estimate, called Aspect for aspect models [21], that relies on the similarity of g to the clusters $c \in C(\mathcal{G}_d)$ and the likelihood that these clusters contain a high ratio of n-grams that are likely to be used as queries:

$$S_{Aspect}(g) \stackrel{def}{=} \sum_{c \in C(\mathcal{G}_d)} S_{AM}(c) \frac{1}{k} \sum_{g' \in c} sim(g', g). \quad (2)$$

As was the case in Equation 1, we use a rank-to-score transformation to compute $sim(g', g)$: $\frac{1}{rank(g; \mathcal{L}_{g'; sim}(\mathcal{G}_d)) + \nu}$; $\mathcal{L}_{g'; sim}(\mathcal{G}_d)$

[4] In our setting, a document essentially plays the role of the query used for ranking document clusters in Raiber and Kurland [38].

is the ranking of all the n-grams $g'' \in \mathcal{G}_d$ induced with respect to g' using $sim(g', g'')$. Refer back to Section 3.2 for details. The Interf method then linearly interpolates, using a parameter λ, the normalized initial (Equation 1) and aspect (Equation 2) scores:

$$S_{Interf}(g) \stackrel{def}{=} (1-\lambda)\frac{S_{Init}(g)}{\sum_{g' \in \mathcal{G}_d} S_{Init}(g')} + \lambda \frac{S_{Aspect}(g)}{\sum_{g' \in \mathcal{G}_d} S_{Aspect}(g')}; \quad (3)$$

that is, g is rewarded if it is highly ranked by f_{init} or if it is similar to n-grams in clusters that contain a high fraction of n-grams that are highly ranked by f_{init}.

Non Cluster-Based Approaches. An alternative approach to Interf that does not utilize clusters is **Top**, which rewards n-grams if they are highly ranked by f_{init} or if they are similar to other highly ranked n-grams which do not necessarily belong to the same clusters. Specifically, let $\mathcal{L}^{\tau}_{f_{init}}(\mathcal{G}_d)$ denote the τ most highly ranked n-grams in $\mathcal{L}_{f_{init}}(\mathcal{G}_d)$; τ is a free parameter. Top interpolates the similarity-based score:

$$S_{Sim}(g) \stackrel{def}{=} \sum_{g' \in \mathcal{L}^{\tau}_{f_{init}}(\mathcal{G}_d)} S_{Init}(g') sim(g', g) \quad (4)$$

with g's initial score:

$$S_{Top}(g) \stackrel{def}{=} (1-\lambda)\frac{S_{Init}(g)}{\sum_{g' \in \mathcal{G}_d} S_{Init}(g')} + \lambda \frac{S_{Sim}(g)}{\sum_{g' \in \mathcal{G}_d} S_{Sim}(g')}. \quad (5)$$

Note that both Top and Interf reward an n-gram based on its initial score and its similarity to other n-grams. In Top, similarity is with respect to n-grams which are initially highly ranked. In contrast, in Interf, similarity is with respect to n-grams which are members of n-gram clusters which are (initially) highly ranked.

The Recursive Weighted Influx method (**RWI**) [23] is based on a weighted directed graph $G = (\mathcal{G}_d, wt)$. An edge is drawn from g ($\in \mathcal{G}_d$) to g' ($\in \mathcal{G}_d \setminus g$) if g' is among the δ nearest neighbors of g. The inter n-gram similarity measure serves to determine the nearest neighbors of an n-gram and the weights of the edges connecting the vertices:

$$wt(g, g') = \begin{cases} sim(g, g') & \text{if } rank(g'; \mathcal{L}_{g;sim}(\mathcal{G}_d)) \leq \delta, \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

Then, the PageRank algorithm [6] with the dumping factor φ is used to estimate the centrality of g:

$$Cent(g) \stackrel{def}{=} \sum_{g' \in \mathcal{G}_d} Cent(g')\Big(\frac{1-\varphi}{|\mathcal{G}_d|} + \varphi \frac{wt(g, g')}{\sum_{g'' \in \mathcal{G}_d} wt(g, g'')}\Big). \quad (7)$$

The final score of g is determined by integrating the initial score in Equation 1 with the centrality score in Equation 7:

$$S_{RWI}(g) \stackrel{def}{=} S_{Init}(g) Cent(g). \quad (8)$$

Thus, g is rewarded if it is initially highly ranked and if its centrality PageRank value in the similarity graph is high. There are two fundamental differences between RWI and the Top and Interf methods. The first is the way the initial n-gram score is used. In Top and Interf, a linear interpolation is used, while in RWI a non parametrized product is used. The second difference is the similarity evidence that is used. In Top and Interf, direct similarity to other n-grams is utilized. In RWI, a more evolved notion of similarity (induced centrality) is utilized.

Table 2: Features used to represent a pair of an n-gram g and a document (article) d for the initial n-grams ranking (Init).

	Feature	Description
Document Independent	IsEnt	Is g an entity? [19]
	ContainEnt	Does g contain an entity? [19]
	Len	Number of terms in g [9, 24]
	UniqLen	Number of unique terms in g [9]
	IMaxLen	Inverse length of g's longest term [9]
Document Dependent	InTitle	Does g appear in d's title? [9, 19]
	IPos	Inverse position of g's first occurrence in d [19]
	TitleOverlp	Fraction of g's terms that appear in d's title [9, 19]
	EntOverlap	Fraction of g's terms that appear in any entity in d [19]
	EntTF	Frequency of g in d if IsEnt = 1; 0 otherwise [19]
	TitleEntTF	Frequency of g in d's title if IsEnt = 1; 0 otherwise [19]
	LogTF	Logarithm of the frequency of g in d [9, 19, 24]
	IDF	IDF of g [9, 19, 24]
	NgramTFIDF	LogTF × IDF [9, 19]

4 EVALUATION

4.1 Experimental Setting

Data. The data used for experiments was collected from the anonymous Yahoo! query log during 50 consecutive days in April-May 2018 in the US. The documents in which we recommend queries are news articles from Yahoo! News as they were shown to trigger many search queries [19, 37]. Our data includes article-query pairs where a query was issued to the search engine by at least three users in a day during the first five minutes after reading the article; no other actions were performed in between by the users. We filtered out 40 navigational queries; e.g., "facebook.com".

Overall, we collected 588590 article-query pairs (15967 articles).[5] Since our goal is to highlight n-grams in the article, we reduced this set to include only pairs in which the query appeared in its entirety in the article. The resultant set, denoted **Base**, contains 26217 pairs (10256 articles). In addition, a random sample of 1000 articles and their corresponding queries (5464 pairs) from Base was annotated by professional in-house editors.[6] The editors were asked to mark pairs in which the query (n-gram) was likely to be "triggered" by (and relevant to) the article (cf. [19]). Each pair was annotated by one editor. This set is denoted **Editorial**. We tokenized the texts, removed punctuation marks and stopwords (on the INQUERY list [1]) from articles and queries. We split the articles into sentences before creating the pool of candidate n-grams to avoid using n-grams that span several sentences. The Stanford CoreNLP toolkit [31] was used for experiments.

Initial Ranking. We apply a learning-to-rank approach to induce an initial ranking of n-grams in an article by their presumed likelihood to serve as queries. The approach does not account for the relations between n-grams. The initial n-gram ranking, henceforth **Init**, is used by the n-gram ranking methods described in Section 3.3 and serves as our baseline ranking from which the initial list $\mathcal{L}_{f_{init}}(\mathcal{G}_d)$ is derived. To induce the initial ranking, each pair of an article and n-gram is represented by a feature vector. In previous work, query logs and personal historical information were used to derive some of the features. These sources of information are not used in our setting. To the best of our knowledge, there is

[5] The inverse document frequency (IDF) and the pre-retrieval predictors in our experiments were computed with respect to this set of articles.

[6] Each article in this set has at least one n-gram in the pool that was used as a query.

no baseline that suits the specific setting we address here. Therefore, we used *all* the features from Kong et al. [19] which do not rely on personal information and additional document-independent features from Cheng et al. [9]. We modified some of the features to rely on corpus statistics rather than on user preferences or historical statistics. A subset of the features we use was shown by Kong et al. [19] to be highly effective for recommending queries that appear in search logs. Hence, our initial ranking is an effective query recommendation baseline.

The features used are listed in Table 2. Some of the features quantify surface-level properties, e.g., Len, UniqLen and IMaxLen[7]. Other features quantify properties related to entities, such as IsEnt and ContainEnt. In addition to the five document-independent features just mentioned, we also use nine document-dependent features. InTitle and IPos are based on the position of the n-gram in the article; these features were also used to learn the inter n-gram similarity function in Section 3.2. TitleOverlp[8] is another feature that gives special emphasis to the article's title. EntOverlap[9], EntTF and TitleEntTF quantify properties related to entities in the context of the article. LogTF, IDF[10] and their product NgramTFIDF, quantify the importance of an n-gram in the article; NgramTFIDF was also used in Section 3.2 for learning the similarity function.

The CLTR method, based on ranking n-gram clusters (see Section 3.3), uses the average and maximum of the values of features from Table 2 attained for the cluster's constituent n-grams. Overall, it is based on 28 features: the 14 features in Table 2 $\times$ 2 (average and maximum).

Evaluation Metrics. We use NDCG@m, MAP@m, P@m (precision) and MRR@m to estimate the effectiveness of the ranked lists of recommended n-grams. All the metrics are computed for the top $m = 5$ and top $m = 20$ ranked n-grams.

In addition, we apply Voorhees' nearest-neighbor cluster hypothesis test [49] to study the extent to which the n-gram cluster hypothesis we proposed (Section 3.2) holds for the similarity function learned in Section 3.2 and other similarity functions used as baselines. For each n-gram $g \in \mathcal{G}_d$ that was used as a query by users reading d, we count how many of its $k-1$ most similar n-grams (neighbors) in d[11] were also used as queries by users.

Statistical significant differences are measured using the two-tailed paired t-test at a 95% confidence level with Bonferroni correction applied to multiple hypothesis testing.

Training Models. We used three effective learning-to-rank [27] approaches for (i) learning the inter n-gram similarity measure, (ii) inducing the initial ranking Init and (iii) training CLTR: RankSVM [18] (SVM in short), MART [13] and LambdaMART [50] (LMART in short).[12] All features were min-max normalized.

We used ten-fold cross validation over articles to train the models and set free-parameter values via the following two-phase procedure. We first used six folds to train a model and three folds to set its hyper-parameter values (i.e., validation). Once the best parameter values were selected, a final model was learned using all nine training folds. This procedure was repeated ten times, where each time a different fold was used for testing. We report the average performance over all the articles when these were part of the test folds. (Each article was a member of a single test fold.)

To learn the n-gram similarity measure, NDCG@4 serves as the optimization metric. (The number of nearest neighbors used for an n-gram is 4.) To rank clusters in CLTR, the cluster score is set to the number of n-grams in the cluster that were used as queries by users.[13] NDCG@20 was the optimization criterion for training Init and CLTR, and for setting hyper parameters.

Free-Parameter Values. The size of the clusters, k, is set to 5. Such small (overlapping) clusters were shown to be effective in past work on cluster-based document [20, 22, 38] and entity [41] retrieval. The Word2Vec and ESA vectors are based on a dump of Wikipedia from July 2018. For Word2Vec, we used the Continuous Bag-of-Words (CBOW) model and set the dimension of the vectors and the window size to 300 and 5, respectively. For ESA, we used Apache Lucene's[14] Okapi BM25 and set $k = 1.2$ and $b = 0.75$. The value of c in SVM was selected from $\{0.1, 0.01, 0.001\}$. The number of trees and leaves in MART and LMART were set to values in $\{100, 250, 500\}$ and $\{10, 25\}$, respectively. For the Proximity feature, we set $\sigma = 2000$ following work on utilizing term proximities in passage retrieval [8]. The values of λ and ν were selected from $\{0, 0.1, \ldots, 1\}$ and $\{0, 30, 60, 90\}$, respectively. The number of top ranked n-grams, τ, in Top is in $\{5, 10, 25, 50\}$. The dumping factor, φ, and the number of nearest neighbors, δ, in RWI are in $\{0, 0.1, \ldots, 0.9\}$ and $\{4, 9, 19\}$, respectively.

4.2 Experimental Results

Our approach is based on creating a pool of candidate n-grams from a document, and recommending n-grams from the pool as queries by ranking the pool. In Section 4.2.1 we demonstrate the merits of the pool creation method proposed in Section 3.1. For this analysis, we use the Base set of article-query pairs. The rest of the analysis and evaluation is based on the Editorial set. In Section 4.2.2 we analyze properties of the inter n-gram similarity measure. In Section 4.2.3 we analyze the performance of our query recommendation methods, which utilize the n-gram similarity measure.

4.2.1 Creating a Pool of Candidate Queries (N-Grams). In what follows, we study the merits of applying the approach proposed in Section 3.1 for creating a pool of n-grams that are candidates for queries. We also provide some insights about the queries that users reading an article issue. The analysis is based on the Base set of query-article pairs. We found that 75% of the queries issued by users reading an article are sequences of entities: 73.7% of these queries contain a person, 11.7% a location and 16% an organization.[15] In addition, 77.2% of all the queries are proper nouns and 60.3% are noun

[7] IMaxLen was originally defined as the maximum length of a term in a query [9]; we use the inverse of the maximum length of a term in an n-gram.
[8] Kong et al. [19] used the term overlap between the query and the title. We use the fraction of terms that appear in the title so as not to favor long n-grams.
[9] Kong et al. [19] used the term overlap between a query and the entities that appear in the article. We count the number of terms in an n-gram that appear in any entity in the article and divide by the n-gram's length so as not to favor long n-grams.
[10] Kong et al. [19] computed LogTF and IDF using a query log.
[11] All the articles in our experiments contain at least k n-grams.
[12] For SVM, we used the SVM^{rank} library (http://svmlight.joachims.org). For MART and LMART, we used the RankLib library (https://tinyurl.com/ranklib).
[13] Using NDCG@5 as the cluster score (cf., [38]) resulted in inferior performance.
[14] http://lucene.apache.org
[15] The numbers do not sum to 100 because an n-gram can contain several entities.

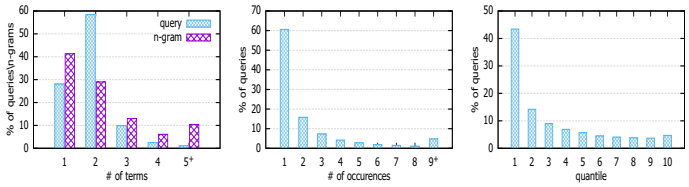

Figure 2: Left: length distribution of queries issued by users and of candidate n-grams; '5+': at least 5 terms. Middle: occurrence distribution of queries in an article; '9+': at least 9 occurrences. Right: position distribution of the first occurrence of queries in articles issued by users; articles are split to ten quantiles.

Table 3: The nearest-neighbor cluster hypothesis test. 'r' and 'l' mark statistically significant differences with Random and LMART (except for Oracle), respectively.

Random	SVM	MART	LMART	Oracle
0.162	0.826^{rl}	1.401^{rl}	1.472^{r}	3.300

phrases. Overall, the candidate pool of n-grams we created includes 90.1% of all the queries in the Base set that were issued by users; for only 3% of the articles there was no intersection between the set of queries issued by users and the candidate n-grams used. The size of the candidate pool is 5.2 times smaller than the total number of n-grams that contain up to 4 terms. These findings suggest that our pool creation approach considerably reduces the number of candidate n-grams, but the pool includes the vast majority of the queries issued by users for the considered articles.

Figure 2 (left) presents the length distribution of the candidate n-grams in the created pool and the queries issued by users reading an article in the Base set. We see that both are relatively short: about 86% of the queries issued by users and 70% of the candidate n-grams in the pool are shorter than 3 terms. This finding is aligned with past work that reported an average query length of 1.9 terms [16]. We also found that the issued queries are not only very short but that they also occur very few times in the article's text as presented in Figure 2 (middle): about 61% of the queries appear only once and about 16% only twice.

Figure 2 (right) presents the position distribution of the first occurrence of queries in an article issued by users; the articles are split into 10 quantiles. We see that users tend to issue queries that appear at the very beginning of the article. This finding echoes the results reported by Kong et al. [19] for queries "triggered" by the article. We also found that 32% of the queries appear in the article's title. Indeed, we show in Section 4.2.2 that InTitle is the most important feature among those considered for learning the inter n-gram similarity measure.

4.2.2 Learning an Inter N-Gram Similarity Measure. Our query recommendation approach utilizes an inter n-gram similarity measure which we now turn to analyze.

Cluster hypothesis test. Table 3 reports the extent to which our cluster hypothesis from Section 3.2 holds according to the nearest-neighbor test described in Section 4.1; the similarity function in the test is learned using SVM, MART and LMART. We present for reference two additional baselines. The first is **Random**, where an n-gram's neighbors are randomly selected[16]. The second is **Oracle**, where the neighbors are selected based on whether they were actually used as queries by users; i.e., this is the upper bound on the nearest-neighbor cluster hypothesis test.

Table 3 shows that the cluster hypothesis always holds to a substantially and statistically significantly larger extent when using the learned similarity function than in Random. The numbers attained for non-liner learning-to-rank models (MART and LMART) are considerably higher than those for a linear model (SVM). The hypothesis holds to the largest extent for LMART: this result (1.472) constitutes 44.6% of the maximum potential (3.3 for Oracle) compared to only 4.9% for Random. This finding suggests that our approach to learning inter n-gram similarities is quite effective in coupling n-gram pairs where both are either used, or not, as queries.

Analyzing SERPs. We next turn to analyze the relations between search results pages (SERPs) retrieved for pairs of similar — as deemed by our learned similarity measure — queries that originate from the articles and their clicks. This type of analysis for article-based query recommendation is novel to this study. We consider query pairs where the first is a *seed*: an n-gram in the pool which was used as a query; the second is any of its $k-1$ *neighbor*s which was also used as a query; for seeds we consider only queries with at least one neighbor which is also a query.

SERP similarity. We measure the similarity between two SERPs using the cosine between the TF.IDF vectors representing the concatenation of the titles of the top-5 Web pages on the SERPs[17]. We then average these similarities for seed-neighbor pairs over search sessions, neighbors per seed and then seeds. The resultant average cosine value for using LMART, MART and SVM to learn the inter n-gram similarity measure which determines nearest neighbors was .164, .161 and .229, respectively; that is, using SVM results in a similarity measure that is more aligned with similarities between retrieved SERPs than when using LMART and MART.

Clicks. For each seed (query) in an article, and for each of its (query) neighbors, we computed the average number of clicks over query sessions — referred to as a click value. We computed Pearson correlation, over articles in the Base set, between the average click value per article of seeds and the average click values for their neighbors: .374, .328 and .411 for LMART, MART and SVM, respectively. That is, using SVM to learn our n-gram similarity measure results in stronger correlation of click rates for the retrieved SERPs of seeds and their neighbors than when using LMART and MART.

To shed more light on the different findings from above between using SVM and LMART or MART, we computed some additional similarity statistics between (query) seeds and their (query) neighbors. The average term-based Jaccard coefficient between a seed and its neighbors was .079, .079 and .161 for LMART, MART and

[16] We compute the probability that a randomly selected neighbor is used as a query (cf., [43]).

[17] IDF was computed over all the SERPs retrieved for all the queries in the Base set.

Table 4: The nearest-neighbor cluster hypothesis test when using a single feature (Only) or excluding the feature (Exclude) to learn the n-gram similarity measure. '$*$' marks statistically significant differences with using all the features.

Family	Feature	Only	Exclude
Textual Similarities	TFIDF	0.425^*	1.432^*
	SentTFIDF	0.336^*	1.446^*
Semantic Similarities	W2V1	0.419^*	1.447^*
	W2V2	0.446^*	1.441^*
	ESA	0.384^*	1.435^*
Proximity Features	SharedSent	0.517^*	1.445^*
	Proximity	0.644^*	1.371^*
N-gram Priors	MaxIDF	0.252^*	1.449
	AvgIDF	0.240^*	1.492
	MaxSCQ	0.706^*	1.354^*
	AvgSCQ	0.619^*	1.398^*
	InTitle	0.821^*	1.368^*
	IPos	0.483^*	1.456
	NgramTFIDF	0.629^*	1.325^*
N-gram Independent Features	DocLen	–	1.442
	Entropy	–	1.461
All Features		1.472	–

SVM, respectively. We also found that the majority of (query) neighbors of a (query) seed that result from using SVM are also neighbors when using LMART and MART, but the reverse does not hold. We thus arrive at the following conclusion: using SVM results in an inter n-gram similarity measure that rewards high surface-level similarities to a higher extent than LMART and MART; this translates to higher SERP and click-rates similarities. On the other hand, using LMART and MART result in improved coupling of n-grams which are queries used by users as the cluster hypothesis results from Section 4.2.2 show. Below we report results when using LMART which yields the highest cluster hypothesis results.

Feature Importance. The similarity function we learn is based on 16 different features. In Table 4 we present the extent to which the nearest-neighbor cluster hypothesis holds when the similarity between n-grams is determined using only one feature at a time. We also show the results when excluding each feature at a time when learning the similarity function. We see that among all the considered features, when used alone, the cluster hypothesis holds to the largest extent for the InTitle feature (0.821) followed by the pre-retrieval predictor MaxSCQ (0.706). These numbers are however much lower than that attained when all the features are used (1.472). In Section 4.2.3 we study the effect of using InTitle instead of our learned similarity function in the RWI n-gram ranking method.

Only the removal of AvgIDF improves the results of the test although not to a statistically significant degree. This finding shows that our similarity function can potentially be further improved by feature selection. We note that AvgIDF, when used alone, is a rather weak similarity indicator. The biggest drop is attained when NgramTFIDF and MaxSCQ are excluded. More generally, when excluding a single feature, we see a statistically significant drop in the extent to which the hypothesis holds, according to the nearest-neighbor test, for 11 out of the 16 considered features. This finding attests to the complementary nature of the features we use.

4.2.3 Recommending Queries. We now turn to analyze the effectiveness of the different n-gram ranking methods we proposed in Section 3.3. Ranking is based on the presumed likelihood that an n-gram will be used as a query. For this evaluation, we use the Editorial set. Unless otherwise specified, we do not use the editors' annotations but rather the information about which recommended n-grams were actually used as queries.

Table 5: Main result: ranking n-grams. 'i' and 'r' mark statistically significant differences with Init and RWI, respectively. Boldface: best results in a column.

	NDCG		MAP		p		MRR	
	@20	@5	@20	@5	@20	@5	@20	@5
Init	58.4	49.5	37.6	30.2	15.4	33.9	71.5	70.4
GM	58.1_r	50.5_r	37.7_r	31.0_r	15.0^i_r	34.4_r	72.6_r	71.5_r
AM	58.3_r	50.3_r	37.7_r	30.7_r	15.2_r	34.1_r	72.3_r	71.3_r
CLTR	55.8^i_r	50.0_r	36.4_r	30.8_r	13.5^i_r	32.7_r	74.2^i	73.1^i_r
Interf	$\mathbf{62.3}^i$	53.3^i_r	41.1^i	33.2^i	$\mathbf{16.2}^i$	35.7^i_r	75.9^i	75.0^i
Top	62.0^i	53.5^i	41.0^i	33.3^i	16.1^i	36.2^i	75.1^i	74.2^i
RWI	$\mathbf{62.3}^i$	$\mathbf{54.3}^i$	$\mathbf{41.2}^i$	$\mathbf{33.8}^i$	16.1^i	$\mathbf{36.8}^i$	$\mathbf{76.1}^i$	$\mathbf{75.3}^i$

Main Result. As already noted, we used three learning-to-rank approaches (SVM, MART and LMART) to induce an initial ranking Init upon which the methods operate. Table 5 presents for each method, including Init, the best results attained when using one of the three approaches[18]. In all cases, the inter n-gram similarity function was learned using LMART. (See Section 4.2.2 for details.)

We see in Table 5 that GM and AM outperform Init in most cases, but the differences are not statistically significant. CLTR, which utilizes aggregates of the features used by Init, outperforms Init in half of the cases (two of these differences are statistically significant); it is also statistically significantly outperformed by Init in two cases. Interf, which uses information induced from clusters to enrich the representation of n-grams, outperforms all the cluster-ranking approaches (GM, AM and CLTR). Interf, Top and RWI always statistically significantly outperform Init, but RWI does so to a larger extent. This finding attests to the merits of utilizing inter n-gram relations, specifically our learned n-gram similarity measure, for ranking n-grams by the presumed likelihood that they will be used as queries. In the majority of the cases RWI outperforms Interf and Top; some of the differences with Interf are statistically significant. Therefore, in the following analysis we focus on RWI.

Varying Free-Parameter Values. In Figure 3 we study the effect of varying the values of the three free parameters incorporated in RWI on its NDCG@20 performance. (Recall that throughout the evaluation, the values are set using cross validation.) We see that the best performance is attained when a relatively large number of neighbors (δ; Equation 6) is used and when a relatively high weight is placed on the inter n-gram similarity estimate which serves as an edge weight ($\varphi > 0.5$; Equation 7). We also see that using a more flat rank-to-score distribution (higher ν in Equation 1) results in reduced performance.

Varying the N-Gram Similarity Function. In work on utilizing inter-document similarities for document retrieval, it was shown that higher results in the nearest-neighbor cluster hypothesis test do not always translate to improved retrieval effectiveness [39]. In Table 6 we present the effectiveness of RWI when the inter n-gram

[18]The best performance of Init, GM and CLTR was attained when LMART was used. The performance of Interf, AM, Top and RWI was best when MART was used.

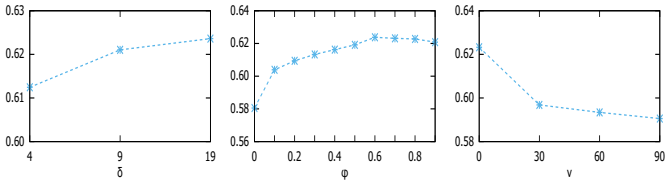

Figure 3: NDCG@20 performance of RWI as a function of δ (left), φ (middle) and ν (right).

Table 6: The effect of the inter n-gram similarity measure on the performance of RWI. 'i' and 'l' mark statistically significant differences with Init and LMART, respectively. The best result in a column is boldfaced.

		NDCG		MAP		p		MRR	
		@20	@5	@20	@5	@20	@5	@20	@5
RWI	Init	58.4	49.5	37.6	30.2	15.4	33.9	71.5	70.4
	MART	59.9^{i}_{l}	51.3^{i}_{l}	38.9^{i}_{l}	31.4_{l}	15.6_{l}	34.7_{l}	74.0^{i}_{l}	73.0^{i}_{l}
	LMART	$\mathbf{62.3}^{i}$	$\mathbf{54.3}^{i}$	$\mathbf{41.2}^{i}$	$\mathbf{33.8}^{i}$	$\mathbf{16.1}^{i}$	$\mathbf{36.8}^{i}$	$\mathbf{76.1}^{i}$	$\mathbf{75.3}^{i}$

similarity measure is learned using MART and LMART. The results for SVM and InTitle[19], the most informative feature according to the cluster hypothesis test (refer back to Section 4.2.2 for the full analysis), are not presented: for these similarity measures the value of φ (Equation 7) was always set to 0 in the cross validation procedure. Hence, their overall performance was identical to that of Init. We see in Table 6 that the best performance is attained for LMART followed by MART and Init (and accordingly SVM and InTitle).[20] The same performance order of approaches was induced by the results of the cluster hypothesis test. (Refer back to Table 3.) To conclude, in our setting, there is correspondence between the extent to which the cluster hypothesis test holds and the resultant performance of a highly effective approach that utilizes the inter n-gram similarity measure.

Varying the Initial Ranking. In Table 7 we present the performance of RWI when different learning-to-rank approaches are used for producing the initial ranking Init; LMART is used to learn the inter n-gram similarity measure. We see that the worst performance of Init is attained for SVM. Yet, we found that the performance attained for SVM (NDCG@20 is 39.7) is much higher than that attained when any feature that it incorporates is used alone: the best results were attained for EntTF (NDCG@20 is 35.1), followed by TitleOverlp (NDCG@20 is 31.5) and InTitle (NDCG@20 is 31.4). (The full analysis is omitted as it provides no additional insights.) Init is more effective when LMART is used, while RWI is more effective when MART is used. We also see that regardless of the learning-to-rank model employed, RWI outperforms Init with all of the differences being statistically significant. This finding suggests that RWI is effective when using initial n-gram ranking functions of varying effectiveness.

[19]Since InTitle is a binary feature, we used MaxSCQ, the second most informative feature, for breaking ties. The results of the cluster hypothesis test for the resultant similarity measure is 0.977.

[20]Init is always outperformed by MART, which is always statistically significantly outperformed by LMART.

Table 7: The effect of the learning-to-rank approach used to induce the initial ranking (Init) on RWI's performance. 'i': statistically significant differences with Init per learning-to-rank model. Boldface: the best result in a column.

		NDCG		MAP		p		MRR	
		@20	@5	@20	@5	@20	@5	@20	@5
SVM	Init	39.7	29.8	21.1	15.1	12.2	23.0	45.0	42.9
	RWI	54.2^{i}	46.6^{i}	34.1^{i}	27.6^{i}	14.2^{i}	32.3^{i}	67.2^{i}	66.0^{i}
MART	Init	58.1	49.3	37.3	30.1	15.4	33.6	70.8	69.8
	RWI	$\mathbf{62.3}^{i}$	$\mathbf{54.3}^{i}$	$\mathbf{41.2}^{i}$	$\mathbf{33.8}^{i}$	$\mathbf{16.1}^{i}$	36.8^{i}	$\mathbf{76.1}^{i}$	$\mathbf{75.3}^{i}$
LMART	Init	58.4	49.5	37.6	30.2	15.4	33.9	71.5	70.4
	RWI	62.1^{i}	$\mathbf{54.3}^{i}$	$\mathbf{41.2}^{i}$	$\mathbf{33.8}^{i}$	$\mathbf{16.1}^{i}$	$\mathbf{37.1}^{i}$	75.8^{i}	74.9^{i}

Table 8: Editorial-based evaluation. 'i': statistically significant difference with Init. Bold: best result in a column.

	NDCG		MAP		p		MRR	
	@20	@5	@20	@5	@20	@5	@20	@5
Init	61.1	51.5	42.3	35.7	12.9	31.2	69.7	68.5
RWI	$\mathbf{65.3}^{i}$	$\mathbf{56.4}^{i}$	$\mathbf{46.6}^{i}$	$\mathbf{40.1}^{i}$	$\mathbf{13.6}^{i}$	$\mathbf{33.9}^{i}$	$\mathbf{74.2}^{i}$	$\mathbf{73.3}^{i}$

Evaluation Based on Editors' Labels. Inspired by Kong et al.'s [19] analysis of queries "triggered" by an article, we study the effectiveness of our approach in predicting queries (n-grams), within the article, that according to human annotators are triggered by (and related to) the article's content[21]. The labels assigned by the editors are used here — and only here — only for evaluation and not for training the models. The results are presented in Table 8. We see that RWI substantially and statistically significantly outperforms Init which, as noted in Section 4.1, is a highly effective baseline for document-based query recommendation.

5 CONCLUSIONS AND FUTURE WORK

We addressed the novel task of recommending n-grams in a document read by a user as search engine queries. Our approach is based on utilizing inter-query relations. More specifically, we presented a new supervised inter n-gram similarity measure, and used it in a variety of query recommendation methods adapted from work on ad hoc document retrieval.

Empirical evaluation attests to the clear merits of using inter n-gram similarities for query recommendation. We also presented a novel type of analysis of document-based recommended queries and applied it to queries used by Yahoo!'s search engine users after they have read documents. The analysis is based on the similarities between SERPs retrieved for the queries and their click rates.

Automatically determining the number of n-grams to highlight in an article as well as deciding which occurrence, or occurrences, of an n-gram to highlight are interesting research challenges. In addition, integrating the preferences and interests of the user with our inter n-gram similarity measure is an interesting future direction.

Acknowledgments. We thank the reviewers for their comments. This paper is based upon work supported in part by the Israel Science Foundation under grant no. 1136/17.

[21]In contrast, the evaluation thus far was based only on whether an n-gram was actually used as a query or not.

REFERENCES

[1] James Allan, Margaret E. Connell, W. Bruce Croft, Fang-Fang Feng, David Fisher, and Xiaoyan Li. 2000. INQUERY and TREC-9. In *Proc. of TREC*. 551–562.

[2] Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. 2004. Query recommendation using query logs in search engines. In *Proc. of EDBT*. 588–596.

[3] Sumit Bhatia, Debapriyo Majumdar, and Prasenjit Mitra. 2011. Query suggestions in the absence of query logs. In *Proc. of SIGIR*. 795–804.

[4] Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, and Sebastiano Vigna. 2009. Query suggestions using query-flow graphs. In *Proc. of WSCD*. 56–63.

[5] Ilaria Bordino, Gianmarco De Francisci Morales, Ingmar Weber, and Francesco Bonchi. 2013. From machu_picchu to rafting the urubamba river: anticipating information needs via the entity-query graph. In *Proc. of WSDM*. 275–284.

[6] Sergey Brin and Lawrence Page. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. In *Proc. of WWW*. 107–117.

[7] David Carmel, Yaroslav Fyodorov, Saar Kuzi, Avihai Mejer, Fiana Raiber, and Elad Rainshmidt. 2019. Enriching News Articles with Related Search Queries. In *Proc. of WWW*. 162–172.

[8] David Carmel, Anna Shtok, and Oren Kurland. 2013. Position-based contextualization for passage retrieval. In *Proc. of CIKM*. 1241–1244.

[9] Zhicong Cheng, Bin Gao, and Tie-Yan Liu. 2010. Actively predicting diverse search intent from user browsing behaviors. In *Proc. of WWW*. 221–230.

[10] Gordon V. Cormack, Charles L. A. Clarke, and Stefan Büttcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proc. of SIGIR*. 758–759.

[11] Van Dang and Bruce W Croft. 2010. Query reformulation using anchor text. In *Proc. of WSDM*. 41–50.

[12] Bruno M Fonseca, Paulo Golgher, Bruno Pôssas, Berthier Ribeiro-Neto, and Nivio Ziviani. 2005. Concept-based interactive query expansion. In *Proc. of CIKM*. 696–703.

[13] Jerome H. Friedman. 2001. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics* 28, 5 (2001), 1379–1389.

[14] Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proc. of IJCAI*, Vol. 7. 1606–1611.

[15] Claudia Hauff, Vanessa Murdock, and Ricardo A. Baeza-Yates. 2008. Improved query difficulty prediction for the web. In *Proc. of CIKM*. 439–448.

[16] Bernard J Jansen, Amanda Spink, and Tefko Saracevic. 2000. Real life, real users, and real needs: a study and analysis of user queries on the web. *Information processing & management* 36, 2 (2000), 207–227.

[17] Nick Jardine and Cornelis Joost van Rijsbergen. 1971. The use of hierarchic clustering in information retrieval. *Information storage and retrieval* 7, 5 (1971), 217–240.

[18] Thorsten Joachims. 2006. Training linear SVMs in linear time. In *Proc. of KDD*. 217–226.

[19] Weize Kong, Rui Li, Jie Luo, Aston Zhang, Yi Chang, and James Allan. 2015. Predicting Search Intent Based on Pre-Search Context. In *Proc. of SIGIR*. 503–512.

[20] Eyal Krikon, Oren Kurland, and Michael Bendersky. 2010. Utilizing inter-passage and inter-document similarities for re-ranking search results. *ACM Transactions on Information Systems* 29, 1 (2010).

[21] Oren Kurland. 2009. Re-ranking search results using language models of query-specific clusters. *Journal of Information Retrieval* 12, 4 (August 2009), 437–460.

[22] Oren Kurland and Carmel Domshlak. 2008. A rank-aggregation approach to searching for optimal query-specific clusters. In *Proc. of SIGIR*. 547–554.

[23] Oren Kurland and Lillian Lee. 2010. PageRank without hyperlinks: Structural reranking using links induced by language models. *ACM Transactions on Information Systems* 28, 4 (2010), 18.

[24] Chia-Jung Lee and W Bruce Croft. 2012. Generating queries from user-selected text. In *Proc. of IIiX*. 100–109.

[25] Kyung-Soon Lee, Young-Chan Park, and Key-Sun Choi. 2001. Re-ranking model based on document clusters. *Information Processing and Management* 37, 1 (2001), 1–14.

[26] Or Levi, Ido Guy, Fiana Raiber, and Oren Kurland. 2018. Selective Cluster Presentation on the Search Results Page. *ACM Transactions on Information Systems* 36, 3 (2018), 28:1–28:42.

[27] Tie-Yan Liu. 2011. *Learning to Rank for Information Retrieval*. Springer. I–XVII, 1–285 pages.

[28] Xiaoyong Liu and W. Bruce Croft. 2004. Cluster-Based Retrieval Using Language Models. In *Proc. of SIGIR*. 186–193.

[29] Xiaoyong Liu and W. Bruce Croft. 2008. Evaluating text representations for retrieval of the best group of documents. In *Proc. of ECIR*. 454–462.

[30] Craig Macdonald, Rodrygo L. T. Santos, and Iadh Ounis. 2012. On the usefulness of query features for learning to rank. In *Proc. of CIKM*. 2559–2562.

[31] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proc. of ACL*. 55–60.

[32] Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proc. of EMNLP*. 404–411.

[33] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR* abs/1301.3781 (2013).

[34] Rodrigo Nogueira and Jimmy Lin. 2019. From doc2query to docTTTTTquery. *Online preprint* (2019).

[35] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document Expansion by Query Prediction. *CoRR* abs/1904.08375 (2019).

[36] Adi Omari, David Carmel, Oleg Rokhlenko, and Idan Szpektor. 2016. Novelty based ranking of human answers for community questions. In *Proc. of SIGIR*. 215–224.

[37] Mandar Rahurkar and Silviu Cucerzan. 2008. Predicting when browsing context is relevant to search. In *Proc. of SIGIR*. 841–842.

[38] Fiana Raiber and Oren Kurland. 2013. Ranking document clusters using markov random fields. In *Proc. of SIGIR*. 333–342.

[39] Fiana Raiber and Oren Kurland. 2014. The correlation between cluster hypothesis tests and the effectiveness of cluster-based retrieval. In *Proc. of SIGIR*. 1155–1158.

[40] Fiana Raiber, Oren Kurland, Filip Radlinski, and Milad Shokouhi. 2015. Learning Asymmetric Co-Relevance. In *Proc. of ICTIR*. 281–290.

[41] Hadas Raviv, Oren Kurland, and David Carmel. 2013. The cluster hypothesis for entity oriented search. In *Proc. of SIGIR*. 841–844.

[42] Eilon Sheetrit, Anna Shtok, and Oren Kurland. 2020. A passage-based approach to learning to rank documents. *Information Retrieval Journal* 23, 2 (2020), 159–186.

[43] Eilon Sheetrit, Anna Shtok, Oren Kurland, and Igal Shprincis. 2018. Testing the Cluster Hypothesis with Focused and Graded Relevance Judgments. In *Proc. of SIGIR*. 1173–1176.

[44] Wei Shen, Jianyong Wang, and Jiawei Han. 2014. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering* 27, 2 (2014), 443–460.

[45] Sandeep Subramanian, Tong Wang, Xingdi Yuan, Saizheng Zhang, Yoshua Bengio, and Adam Trischler. 2017. Neural models for key phrase detection and question generation. *CoRR* abs/1706.04560 (2017).

[46] Idan Szpektor, Aristides Gionis, and Yoelle Maarek. 2011. Improving recommendation for long-tail queries via templates. In *Proc. of WWW*. 47–56.

[47] C. J. van Rijsbergen. 1979. *Information Retrieval* (second ed.). Butterworths.

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proc. of NIPS*. 5998–6008.

[49] Ellen M. Voorhees. 1985. The cluster hypothesis revisited. In *Proc. of SIGIR*. 188–196.

[50] Qiang Wu, Christopher J. C. Burges, Krysta Marie Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Information Retrieval* 13, 3 (2010), 254–270.

[51] Ying Zhao, Falk Scholer, and Yohannes Tsegay. 2008. Effective Pre-retrieval Query Performance Prediction Using Similarity and Variability Evidence. In *Proc. of ECIR*. 52–64.