Query Rewrite for Null and Low Search Results in eCommerce

Zehong Tan
eBay Search Science
zetan@ebay.com

Canran Xu
eBay Search Science
canxu@ebay.com

Mengjie Jiang
eBay Search Science
menjiang@ebay.com

Hua Yang
eBay Search Science
hyang2@ebay.com

Xiaoyuan Wu
eBay Search Science
xiaowu@ebay.com

ABSTRACT

In most cases when users encounter with null and low search queries, there are lots of inventory to these queries on eBay, but they can't be retrieved by regular query expansion. Therefore, retrieving more relevant items by performing more aggressive query rewrites will largely improve the user experience. This paper proposes a query rewrite system that reformulate the original query into multiple alternative queries that enlarge recall without altering the main search intent. In this paper, we present a term dropping algorithm that understands and keeps the important terms in the query, and a term replacement algorithm that learns from user behavior and language model. Experiments on randomly sampled eBay queries shows that the n-best rewrites by these two algorithms effectively recover more relevant items. Besides, the experiments indicate that our algorithm can predict the exact user refined query in its top-5 rewrite candidates for over 50% of the null and low queries.

Keywords

eCommerce, Search Experience, Query Rewrite, Language Model

1. INTRODUCTION

When the retrieved items of a query is lower than some certain number (e.g. 10), the search results page results in bad customer experiences. Therefore recovering more items for null and low queries will dramatically increase conversion and click through rates. Causes of this issue range from a misspelled word to a too specific query. Therefore, with a few exceptions this task can be tackled by simple rule-based solutions. In this scenario, dropping or replacing some terms in these queries may recover more items. The current null and low recovery algorithm randomly drops tokens in the user query up to some certain percentage and come up with a collection of subqueries. Subsequently a relevance model will be used to boost the relevant items in the ranking phase. However, randomly dropping or replacing terms may alter the original search intent which could result in retrieving irrelevant items. For example, when a null query "led brake light nano made in taiwan" is sent for term dropping using the current algorithm, top subqueries could preserve unimportant segments like "made in taiwan" As a result, even though these items contain enough number of tokens in the user query, the relevance is low and user experience is bad. To address this issue, we implement separate algorithms that can drop or replace terms using techniques originated from natural language processing. These algorithms will attempt to recognize the important segments of the query so that dropping or replacing terms will not distort its intent.

The task of query term dropping is assisted by tagging the part-of-speech (POS), entities, unit of measures (UOM) and phrases. In the eCommerce domain, entities are item properties such as brand name, material, color, shape, gender, etc. On the other hand, for null and low queries that are not verbose, we will reformulate the query by replacing some of the tokens. This approach avoids accidental turning the specific seed query into a broad query. To achieve this, we build a language model from large amounts of user query logs. With a given set of randomly selected null and low queries, we focus on the evaluation of n-best rewrites that are produced for evaluation.

To summarize, our main contributions are:

- We propose an effective algorithm to drop unimportant terms without distorting the intent of the query
- We apply a language model to estimate the probability of candidate queries for term replacement.

2. RELATED WORKS

Information retrieval with verbose queries has attracted lots of interest in recent years. Redundancies that reside in these queries suggest that generating subqueries by dropping unimportant terms using natural language processing (NLP) techniques[1, 2, 3] in the original query would be helpful. To extract patterns to reduce long queries, researchers tried to extract features like tf-idf, POS, entities and similarity scores to the original queries. When the retrieved documents are available, more sophisticated work employs random walk from each token in the original query on a graph of retrieved documents and the score of each suggestion is ranked by the Hadamard product of each term[4].

In: J. Degenhardt, S. Kallumadi, M. de Rijke, L. Si, A. Trotman, Y. Xu (eds.): *Proceedings of the SIGIR 2017 eCom workshop, August 2017, Tokyo, Japan,* published at http://ceur-ws.org

Query rewrite systems can be traced back to 2001's when Google started to provide dynamic spelling checking systems. Almost simultaneously, Yahoo started to track common misspelling of frequent queries such as movie stars. Technical details for these systems are not available, but they seem to be based on spelling algorithms and statistical frequencies. Traditional query rewriting system tries to use rule-based techniques such as finding synonyms of words that are extracted from wordnet[5]. In recent years, techniques that utilize NLP and statistical models have been widely used for tasks of query rewrite systems. Mays et. al.[6] showed that noisy channel model could be used for both real-word or non-word errors spelling correction. In these works, either a set of real-word confusion pairs or a lexicon against the edit distance is prepared[7]. [8, 9] proposed to use a large set of ordered query pairs that are scored by their semantic similarities. These algorithms are not specifically designed for null and low search queries. On the other hand, [10] tried to add taxa constraints to sub-queries and retrieve items from a prebuilt database of historical purchases of null and low queries. Distribution of inferred taxa constraints from the historical data are then added to each subquery for secondary retrieval. Furthermore, using query-snippet pairs from user query logs enlarges the scope of query expansion in an improved contextual way[11, 12]. Some works[13] tried to build a query rewrite system without requiring a predefined dictionary. There are several other researches that are relevant to query rewrite, such as using large human labeled data set to build a conditional random fields discriminative model[14].

3. QUERY TERM DROPPING

In this section, we describe the proposed approach to query term dropping. The aim of term dropping is to reduce a verbose query to multiple short query candidates to increase recall as well as keep the query intent.

To understand users' intent, we tag the possible product and aspects in the user query, including brands (such as Nike), product names (such as Macbook and Play Station 2), colors, materials, product models (such as hd-650), UOM etc. Entities and attributes are tagged by mapping the dictionaries mined from eBay's structured catalog. As null and low queries are mostly tail queries, most of the tokens cannot be covered by structured data. The syntax functionalities of these tokens are labeled by the POS tagging. Moreover, phrases are also detected in this phase by mining the user query logs to make sure tokens in a phrase will be dropped or kept as a whole. To this end, when there are multiple possible tagging sequences, we select the sequence that tags most of the tokens. This query tagging procedure provides a context for core concept and attributives extraction in term dropping.

The selection of terms to drop is set by predetermined rules. Brands, product names or first noun phrases are considered as Tier 1 group. They are the likely core product in this query. Aspects (colors, materials, etc), UOM and adjectives etc. are considered as Tier 2 group, as they are the attributes of the core product. The stop words and conjunctive words etc. are considered as Tier 3 group. To this end, a term dropping suggestion is generated by the core product terms in combination with attribute terms. Under the requirement of minimum number of tokens of the alternative query, the n-best suggestions are by generated a scoring algorithm that considers both the alternate query length and the quality of the tokens.

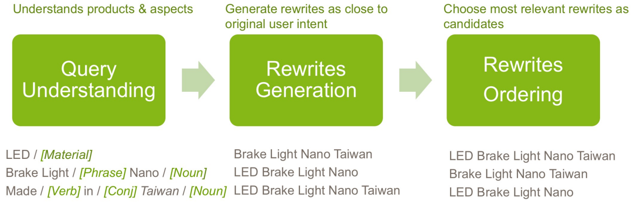
Understands products & aspects
Generate rewrites as close to original user intent
Choose most relevant rewrites as candidates
Query Understanding
Rewrites Generation
Rewrites Ordering
LED / [Material]
Brake Light / [Phrase] Nano / [Noun]
Made / [Verb] in / [Conj] Taiwan / [Noun]
Brake Light Nano Taiwan
LED Brake Light Nano
LED Brake Light Nano Taiwan
LED Brake Light Nano Taiwan
Brake Light Nano Taiwan
LED Brake Light Nano

Figure 1: Workflow of query term dropping. (a) Query understanding phase: each token of the query is tagged by part-of-speech or entities. (b) Rewrites generation phase: drop unimportant tokens to generate subqueries. (c) Rewrites ordering phase: each candidate is assigned with a score according to the tags and passes (see text), and the ranking of the candidates are given accordingly.

3.1 Brand Disambiguation

Due to the ambiguity of brands which are derived from real word, recognition of brands such as "white" or "1928" requires an additional disambiguation algorithm. Therefore we build a discriminative model $P(B|q)$ that predicts the term B is a brand in the context of query q. This is achieved by computing

$$P(B|q) = \sum_i P(C_i|q)P(B|C_i), \qquad (1)$$

where C_i is the ith shopping category. In Eq. (1), $P(C_i|q)$ is the model that projects the context of the query onto the shopping categories, and $P(B|C_i)$ is extracted from the data from eBay's inventory. The model that learns $P(C_i|q)$ is evaluated by inferred category demand. The optimal threshold for classification is determined by the user click data. Tagging procedure for brand name will be performed once Eq. (1) has been run through the entire token list in the query.

3.2 Scoring Algorithm

The alternative query generation is implemented as a multi-pass process. Earlier passes keep more original tokens in the alternate query, resulting in more efficient query with less recalls. Later passes will keep fewer tokens to enlarge the recall size. The search engine issues the term dropping queries pass by pass, and stops whenever it gets enough effective alternative queries and items to show to users. An offline experiment suggests that for 70% of the null and low queries, it need only one pass to complete the whole process. Suggestions from the earlier pass will always rank higher than the ones from later passes, as overall queries with more original tokens are believed to be closer to the original user intent.

In order to score the suggestions in one pass, we basically prefer suggestions with more brands or nouns over suggestions with more adjectives. The baseline tagging score is given by a step function over their tags. After tagging, we found that there are remaining 54% tokens are nouns. Due to the fact that lots of null and low queries have misspelled words that are of low document frequencies, we are inspired to rank these nouns by the cross-entropy of their clicking probabilities conditioning on their document frequencies:

$$P_{\text{click}}(r) = -P(\text{click}|r)\log\left(1 - P(\text{click}|r)\right), \qquad (2)$$

where r is the document frequency of token, and $P(\text{click}|r)$ is evaluated by a large sample of null queries. The clicking score is added to the tagging score with an undetermined coefficient, which is later optimized by maximizing the discounted cumulative gain (DCG).

4. QUERY TERM REPLACEMENT

In this section we describe the proposed approach to term replacement. The inputs are a query q_0 consisting of a sequence of m words $(w_1, \ldots, w_m)$. As the formalism in [6], the algorithm generates a set of candidates $C(q_0)$, and then the best suggestion is given by the highest language model probability. To generate the candidates, we start by generating a set of candidate words for each input word w_i that is mined from the user behavior logs. Thus the probability for each candidate query $q_1 \in C(q_0)$ would be proportional to $P(q_0|q_1)P(q_1)$, where $P(q_0)$ is the probability of the original query modeled by the n-gram language model. Therefore, term replacement is formulated as

$$q_1^* = \arg\max_{q_1 \in C(q_0)} P(q_0|q_1)P(q_1), \qquad (3)$$

where $P(q_0|q_1)$ and $P(q_1)$ is the noisy channel model and language model respectively. Furthermore, in practice, considering the fact that the number of candidates in the noisy channel model of a particular token ranges differently to the cardinality of bigrams, we raise the noisy channel by a power γ:

$$q_1^* = \arg\max_{q_1 \in C(q_0)} P(q_0|q_1)^\gamma P(q_1). \qquad (4)$$

4.1 Noisy Channel Model

The noisy channel model, $P(q_0|q_1)$, describes the emission probability of an original query q_0 being replaced by q_1. We collect query pairs of term replacement within the same session from user query logs. For each original query, we assume that only one token w_i is replaced by $w_i^{'}$, therefore $P(q_0|q_1) = P(w_i|w_i')$.

Note that for each observed token w_i, the probability $P(w_i|w_i' = w_i)$ should be counted as well. Unlike [Mays 1991], where the probability of w_i not being replaced is assigned with a tuning parameter, it is estimated by the effective non-replacement count:

$$P(w_i|w_i' = w_i) = \frac{Count(w_i \rightarrow w_i)}{Count(w_i \rightarrow w_i')}, \qquad (5)$$

where $Count(w_i \rightarrow w_i) = (Count(w_i) - Count(w_i \rightarrow w_i \neq w_i'))/2$.

4.2 Language model

Language modeling in our approach utilizes an n-gram language model that assigns the following probability to a string of words:

$$P(w_1 \ldots w_m) = \prod_{i=1}^{m} P(w_i|w_1^{i-1}) \simeq \prod_{i=1}^{n} P(w_i|w_{i-n+1}^{i-1}) \qquad (6)$$

As shown in [15], language models solely built from query, instead of the mixture of query and others, would be the most predictive. Unlike grammatically strict natural language for long text, the feature of user queries is even more sparse. Therefore we only incorporate bigram features in the language model. Remedies of the sparsity of bigram features are achieved by Kneser-Ney smoothing technique[16].

In the scenario of web search for eCommerce, the terms in the queries are not necessarily following a chronicle order. If a user types “case iphone”, the intent is not far from the query “iphone case”. For this reason, we also incorporate the backward-bigrams in the model without enhancing the number of parameters. Since the backward-bigram probability is not commensurate with the normal bigrams, these probabilities are assigned with a free parameter λ:

$$P(w_1 \ldots w_n) \simeq \prod_{i=1}^{n-1} P(w_{i+1}|w_i) \prod_{i=2}^{n} P(w_{i-1}|w_i)^\lambda \qquad (7)$$

Finally, instead of just choosing a candidate term replacement with the highest probability, we only accommodate the candidates that have higher probability estimated from the language model than the original query. This is to remedy over-replacing issue, that is, avoid null and low queries that are not suitable for term replacement.

5. EXPERIMENT

We test how well our alternative query service can perform on the eBay search engine. The evaluation scenario aims at measuring the ability of the algorithm on recovering null and low search result page and retrieve relevant items.

5.1 Data Preparation

We sampled two query sets of null and low queries from search log for term dropping and replacement separately. The training set for term dropping consists of one week of user's search and click data. The training data for the query term replacement consists of query pairs taken from 11 weeks of query logs. The sampled test set contains 3,000 null and low queries from the consecutive week of the training data.

5.1.1 Query Term Dropping

In the experiment for term dropping, we leverage eBay's structure data dictionary for the product and attribute recognizing in the query tagging phase. When we perform POS tagging for search queries, we employ the Stanford bidirectional model. However, in the scenario of web search, the pior probability of verbs is much less than ordinary new articles, we perform an offline correction on top of the Stanford POS tagging. With mined unigram and bigrams from 9 days' search queries, the tagging for phrases are obtained by the normalized mutual information:

$$MI(w_i, w_{i+1}) = \frac{P(w_i, w_{i+1})^2}{P(w_i)P(w_{i+1})} \qquad (8)$$

5.1.2 Query Term Replacement

We remove all non-alphanumeric characters from the queries or spelling corrections pairs. After filtering, there are 380 millions of query pairs. A collection of data statistics of the training data is shown in Table 1.

Table 1: Statistics of unique n-grams in language model.

	1-gram	2-grams	Noisy Channel
Parameters	2.8 million	29 million	18 million
Count	300 million	300 million	126 million

5.2 Results on Search Metrics

Our algorithm is fine-tuned in order to balance the capability of retrieving more items and their relevances. In this section, we present the result of our experiment on the search metrics.

5.2.1 Recovery of user behaviors

Users' intent is reflected by their own query refinements when they are not satisfied with the returned search result of the original query. We first evaluate the probability of exact predictions of our algorithm from one of the n-best query rewrite candidates. The results are summarized in Table 2.

Table 2: Exact match between user's refinements and n-best query rewrite candidates.

	Top-1	Top-5
Dropping	23.3%	54.6%
Replacement	31.9%	66.4%

5.2.2 Recall Enhancement

In Table 3, we show the number of retrieved items, R, of the original query and the best rewritten query. For comparison, we also show the number for users' rewritten query. We are recovering the null and low search result pages in general.

Table 3: Medium of number of retrieved items, R, of original query (Or) , user's rewritten query (Re) and 3-best query rewrite candidates.

	Or	Re	1st	2nd	3rd
Dropping ($R = 0$)	0	8	8	5	12
Dropping ($R \leq 25$)	2	42	75	45	75
Replacement ($R = 0$)	0	41	3	365	0
Replacement ($R \leq 25$)	4	70	44	362	7

5.2.3 Category Overlap

The distribution of queries over shopping categories is key indicator of high-level relevance for query rewrite systems in eCommerce[10, 17]. It is a sufficiently strong signal that the alternative query distorts user's search intent if the inferred level-2 categories are changed. The category overlap is computed as the Jaccard similarity of up-to top 50 items retrieved from the alternative query, C_A^{50}, to the dominant category of the original query, C_O:

$$J(A, O) = \frac{|C_A^{50} \cap C_O|}{|C_A^{50} \cup C_O|} \tag{9}$$

In Table 4, we show the category overlap of top 3 suggestions by term dropping and term replacement. Overall 71.5% (68.6%) of the items retrieved by top-1 term dropping (replacement) suggestion has the same category with the original query. Short queries with only 2-3 tokens are more troublesome in term dropping with overlap rate 60.4%, as dropping even one term may change the users' intent largely. For example, in the query "boden shoes 30", dropping token "30" (size 30) will alter the category from "kid's clothing and shoes" to "women's shoes". Similar issues can be relieved by term replacement. Our result achieved an overall 74.3% category overlap, compared to 74%, as reported in [10].[1]

Table 4: Category overlap between original query and n-best query rewrite candidates.

	1st	2nd	3rd	overall
Term dropping	71.5%	65.5%	63.3%	66.8%
Term replacement	68.6%	93.1%	83.9%	81.9%

5.3 Qualitative Results

One important feature of our algorithms that lead to better search experiences is the explicit mechanism to rewrite search queries based on the context. In Table 5, we show several examples of top query rewrite generated from the algorithm.

As for term dropping, in the first query "tommy hilfiger" is detected as brand and "sport bra" as an unbreakable noun phrase. Our term dropping rewrites the query by dropping "prep" "free" and misspelled word "shipp". In query "10w high quality speaker 6ohm", our algorithms recognizes that it contains two UOM constraints "10w" "6ohm" and relaxes the query by dropping one of the UOM constraints. As mentioned in section 3.1, the brand names were disambiguated by considering the shopping query. Though word "watt" often refers to a unit at most of the time, it is a high confident brand when it occurs in queries in pottery&craft category. In query "watt spagehetti platter", our algorithm recognizes "watt" as a china brand and keeps it in query rewriting. For term replacement, the query "iphone watch" is rewritten as "apple watch", instead of "iphone clock", because the bigram probability P_b(watch|iphone)is much greater than the emission probability P_e(clock|watch). On the other hand, "amber ball insect" is replaced by "amber sphere insect" because the Kneser-Ney smoothing correctly compensate the probability of "amber sphere", instead of recognizing "amber" as a color.

6. CONCLUSION AND OUTLOOK

In this work, we have implemented algorithms that aim to cure the null and low search results. In particular, we showed that tagging technique developed in NLP could be used to generate more context relevant subqueries, and that additionally statistical natural language model provides a promising avenue for making progress on the path to building full natural language understanding systems. Our nu-

[1] To our knowledge, the most relevant work to ours is Ref[10], in which the authors reported an algorithm to generate search sub-queries in the scenario of null search page with additional taxa constraints. However, the result of this approach is not feasible to reproduce because the expense of inferred exact taxa constrains of each query is maintaining a dynamic database as large as the active inventory. For this reason, this baseline is compared to the reported number in the original literature, instead of reproducing its result for the same query set.

Table 5: Examples of top query rewrite from the algorithms.

	Or	1st
Dropping	tommy hilfiger prep sport bra nwt free shipp 10w high quality speaker 6ohm watt spagehetti platter	tommy hilfiter sport bra nwt 10w high quality speaker watt platter
Replacement	iphone watch amber ball insect	apple watch amber sphere insect

merical experiments indicate that user's shopping experience could be improved with higher level of engagement.

The improvements achieved by our algorithms are not unique to vanilla null and low search pages, and are readily applicable to even broader tasks with search engine, such as extracting structured data from item titles, general query reformulation and anomalous query synopsis. It could lead to significant improvements in shopping experiences with similar models especially with text data of less grammatical strictness.

7. REFERENCES

[1] Kumaran, G., and Carvalho, V. R. Reducing long queries using query quality predictors. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval* (2009), ACM, pp. 564–571.

[2] Chen, Y., and Zhang, Y.-Q. A query substitution-search result refinement approach for long query web searches. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 01* (Washington, DC, USA, 2009), WI-IAT '09, IEEE Computer Society, pp. 245–251.

[3] Duan, H., and Hsu, B.-J. P. Online spelling correction for query completion. In *Proceedings of the 20th International Conference on World Wide Web* (New York, NY, USA, 2011), WWW '11, ACM, pp. 117–126.

[4] Maxwell, K. T., and Croft, W. B. Compact query term selection using topically related text. In *Proceedings of the 36th Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval* (2013).

[5] Zhang, J., Deng, B., and Li, X. Concept based query expansion using wordnet. In *AST '09 Proceedings of the 2009 International e-Conference on Advanced Science and Technology* (2009), pp. 52–55.

[6] Mays, E., Damerau, F. J., and Mercer, R. L. Context based spelling correction. In *Information Processing and Management: An International Journal* (1991), vol. 27, pp. 517–522.

[7] Ahmad, F., and Kondrak, G. Learning a spelling error model from search query logs. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing* (Stroudsburg, PA, USA, 2005), HLT '05, Association for Computational Linguistics, pp. 955–962.

[8] Jones, R., Rey, B., Madani, O., and Greiner, W. Generating query substitutions. In *Proceedings of the 15th International Conference on World Wide Web* (New York, NY, USA, 2006), WWW '06, ACM, pp. 387–396.

[9] Hasan, M. A., Parikh, N., Singh, G., and Sundaresan, N. Query suggestion for e-commerce sites. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining* (New York, NY, USA, 2011), WSDM '11, ACM, pp. 765–774.

[10] Singh, G., Parikh, N., and Sundaresan, N. Rewriting null e-commerce queries to recommend products. In *Proceedings of the 21st International Conference on World Wide Web* (New York, NY, USA, 2012), WWW '12 Companion, ACM, pp. 73–82.

[11] Riezler, S., and Liu, Y. Query rewriting using monolingual statistical machine translation. *Computational Linguistics 36*, 3 (sep 2010), 569–582.

[12] Chen, Q., Li, M., and Zhou, M. Improving query spelling correction using web search results. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (2007), pp. 181–189.

[13] Gao, J., Li, X., Micol, D., Quirk, C., and Sun, X. A large scale ranker-based system for search query spelling correction. In *Proceedings of the 23rd International Conference on Computational Linguistics* (2010), Association for Computational Linguistics, pp. 358–366.

[14] Guo, J., Xu, G., Li, H., and Cheng, X. A unified and discriminative model for query refinement. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (2008), pp. 379–386.

[15] Huang, J., Gao, J., Miao, J., Li, X., Wang, K., Behr, F., and Giles, C. L. Exploring web scale language models for search query processing. In *WWW '10 Proceedings of the 19th international conference on World wide web* (2010), pp. 451–460.

[16] Kneser, R., and Ney, H. Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on* (1995), vol. 1, IEEE, pp. 181–184.

[17] Hasan, S., Heger, C., and Mansour, S. Spelling correction of user search queries through statistical machine translation. In *EMNLP* (2015), L. MÃ rquez, C. Callison-Burch, J. Su, D. Pighin, and Y. Marton, Eds., The Association for Computational Linguistics, pp. 451–460.