Intent Term Weighting in E-Commerce Queries*

Saurav Manchanda†
University of Minnesota
Twin Cities, MN, USA
manch043@umn.edu

Mohit Sharma‡
WalmartLabs
Sunnyvale, CA, USA
sharm163@umn.edu

George Karypis
University of Minnesota
Twin Cities, MN, USA
karypis@umn.edu

ABSTRACT

E-commerce search engines can fail to retrieve results that satisfy a query's product intent because: (i) conventional retrieval approaches, such as BM25, may ignore the important terms in queries owing to their low *inverse document frequency* (IDF), and (ii) for long queries, as is usually the case in rare queries (i.e., tail queries), they may fail to determine the relevant terms that are representative of the query's product intent. In this paper, we leverage the historical query reformulation logs of a large e-retailer (walmart.com) to develop a distant-supervision-based approach to identify the relevant terms that characterize the query's product intent. The key idea underpinning our approach is that the terms retained in the reformulation of a query are more important in describing the query's product intent than the discarded terms. Additionally, we also use the fact that the significance of a term depends on its context (other terms in the neighborhood) in the query to determine the term's importance towards the query's product intent. We show that identifying and emphasizing the terms that define the query's product intent leads to a 3% improvement in ranking and outperforms the context-unaware baselines.

CCS CONCEPTS

• **Information systems** → **Query intent**; **Query reformulation**.

KEYWORDS

Term weighting, query intent, query refinement, query reformulation

ACM Reference Format:
Saurav Manchanda, Mohit Sharma, and George Karypis. 2019. Intent Term Weighting in E-Commerce Queries. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM '19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3357384.3358151

*The code for this paper is available at https://github.com/gurdaspuriya/query_intent
†Work was done when Saurav was a summer intern at WalmartLabs.
‡Work was done when Mohit was at WalmartLabs.

CIKM '19, November 3–7, 2019, Beijing, China

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3358151

1 INTRODUCTION

Retrieving the products that are relevant to a search query in an e-commerce site is a fundamental problem as it is the first step a customer performs during an e-commerce transaction. This is often done by using the search engine to generate a candidate set of a results having few hundreds or thousands of products, and then employing learning-to-rank approaches to generate the final ranked list. To generate a candidate set, search engines use state-of-the-art term frequency based retrieval approaches, such as BM25F [8, 9]. However, e-commerce search engines that rely on these retrieval approaches suffer from two issues. First, a term that frequently occurs in the catalog will have a low *inverse document frequency* (IDF), thereby lowering its contribution in the final score for queries having that term. Such low-contributing terms can affect the recall of products as they can be critical, e.g., the term "women" in "women shoes" is important but due to its high frequency and consequently low IDF, the search engine may show "men shoes" leading to bad customer experience, followed by loss in sales and revenue. Second, when a query has multiple terms, as is usually the case with rare queries (often referred to as *tail queries*), search engines will retrieve all the products whose attributes match the terms in the query, but this can negatively affect the search results as it also matches the terms in the query that do not describe the query's product intent, e.g., the query "stove-top milk steamer for espresso beans coffee maker" may return "coffee beans", "coffee maker", "stove-top coffee maker" and "espresso coffee maker" in addition to "milk steamer". Therefore, identifying the relevant terms in a query that represent its product intent is an important challenge.

In order to identify the terms that express a query's product intent, we do not rely on labeled dataset but instead, we leverage the query reformulation logs to develop a distant-supervised approach [6]. Our approach *contextual term-weighting* (CTW) predicts a weight for each query term that indicates the importance of that term towards defining the query's product intent. As a source of distant-supervision, CTW leverages the fact that the terms in the reformulated query describe the query's product intent better than the terms in the original query. Additionally, CTW is context-aware as it leverages the context of a term to determine its importance. For example, the same term *3-piece* has different significance for the queries like "3-piece kids dinnerware" and "3-piece mens suit". CTW uses Recurrent Neural Networks (RNNs) as the underlying model to model the context.

Although our approach can be applied to all queries, our focus is to evaluate its performance on the rare queries that usually are challenging to work with because of the lack of historical engagement data [2]. The proposed approach improves the ranking performance as measured by the mean reciprocal rank (MRR) [12], of the search results by 3% over the BM25F [9] approach. This improvement is

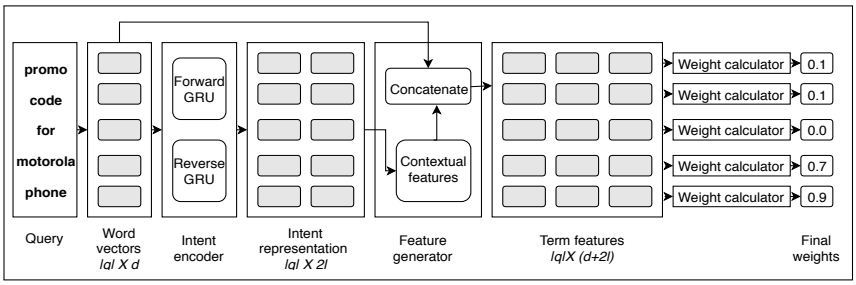

Figure 1: Contextual term-weighting (CTW) model.

significant for a retailer having millions of products in the catalog and will help to generate a better set of initial candidate products followed by the application of Learning-to-Rank methods [4] to produce the final ranking.

2 RELATED WORK

Term-weighting methods: The prior work in this area has focused on ranking the terms according to their discriminating power in determining relevance. Popular examples include term frequency-inverse document frequency (TF-IDF) [10] and Okapi Best Matching 25 (Okapi BM25) [8, 9]. These techniques tend to assign higher weights to the terms that are common in a single or a small group of documents. However, from an e-commerce perspective, a discriminatory term is not necessarily important towards defining that query's product intent [5]. To address this limitation, vector propagation model (VPCG & VG) [3] learns representations of the queries/documents based on the click-graph and use a linear regression model to learn weights for the n-grams in the queries/-documents. However, its limitation is that it is context unaware as it learns a single weight for each n-gram.

Improving relevance for the tail queries: One of the prior approaches for this task predicts the likelihood of one query being the reformulation of another query [2]. This model works well when the set of the candidate reformulations is small, which is not the case in e-commerce. Another class of methods propagate historical engagement information from the most frequent (head) queries to the tail queries based on their common words. One such approach extracts single or multi-word expressions from the head and tail queries and maps them to a common concept space defined by a knowledge-base [11]. However, its limitation is that mapping tail queries to a logical concept space using query text is prone to the presence of irrelevant terms.

3 PROPOSED APPROACH: CONTEXTUAL TERM-WEIGHTING (CTW)

In order to identify the terms that express a query's product intent, we do not rely on access to a labeled data but use the query reformulation logs to develop a distant-supervised approach to find terms that represent the query's product intent. Specifically, we assume that we are given a query q and its reformulation $R(q)$ such that the results displayed for q does not satisfy its product intent while results displayed for $R(q)$ satisfy the product intent. We assume that the terms in q that represent its product intent are also retained in $R(q)$, i.e., the terms in set $q \cap R(q)$ are more important than the terms in set $q \setminus R(q)$. Our approach *contextual term-weighting (CTW)* model solves this problem by predicting a weight for each term in a query that indicates the importance of that term towards defining the query's product intent. CTW leverages Recurrent Neural Networks (RNNs) to capture the context of a term, i.e., the entities in the neighborhood of the term, to estimate its weight. Given a query $\langle v_{q_1}, \ldots, v_{q_{|q|}} \rangle$, CTW estimates a weight $s(v_{q_t})$ for each term v_{q_t}, which captures how important is v_{q_t} towards defining q's product intent. The supervision during training focuses that the terms in the set $q \cap R(q)$ are more important in defining the product intent than the terms in the set $q \setminus R(q)$.

Figure 1 provides an illustration of the contextual term-weighting model. The contextual term-weighting model has three modules: *intent encoder*, *feature generator* and *weight calculator*. The *intent encoder* takes q as input, and outputs a representation of the product intent of q. Next, the *feature generator* uses the output of the *intent encoder* as input and generates the features for each query term v_{q_t}, which captures its importance towards the product intent of q. For each term v_{q_t}, the *weight calculator* uses the features produced by the *feature generator* as input and outputs the importance weight for it. We discuss these three modules in detail below and further details are provided in [5]:

Intent encoder: This module encodes the sequence of terms in the query q into a fixed length representation using bidirectional Gated Recurrent Units (GRUs) [1]. GRUs are a type of Recurrent Neural Networks (RNNs), which model variable-length sequential input using a recurrent, shared hidden state. The hidden state can be thought of a summary of the complete sequential input. The sequence of word vectors in the query q is denoted by $\langle \mathbf{r}_{v_{q_1}}, \ldots, \mathbf{r}_{v_{q_{|q|}}} \rangle$. The bidirectional GRU encodes the query q as:

$$\mathbf{h}_t^f = GRU^f(\mathbf{h}_{t-1}^f, \mathbf{r}_{v_{q_t}}); \quad \mathbf{h}_t^b = GRU^b(\mathbf{h}_{t+1}^b, \mathbf{r}_{v_{q_t}}),$$

where GRU^f encodes the query q in the forward direction and GRU^b encodes in the backward direction. $\mathbf{h}_t^f$ is the hidden state at position t for the GRU^f and corresponds to the summary of the sequence $\langle \mathbf{r}_{v_{q_1}}, \ldots, \mathbf{r}_{v_{q_t}} \rangle$. Similarly, $\mathbf{h}_t^b$ is the hidden state at position t for the GRU^b and stores the summary of the sequence $\langle \mathbf{r}_{v_{q_t}}, \ldots, \mathbf{r}_{v_{q_{|q|}}} \rangle$. The output of the intent encoder are the hidden states ($\mathbf{h}_t^f$ and $\mathbf{h}_t^b$) at each position t.

Feature generator: This module generates features for each term v_{q_t} in the query q, the features capturing its importance towards defining the q's product intent. The importance of a term is not just dependent on the term itself, but also on the context in which the term is used. For example, the same term *3-piece* can have different importance for the different queries like "3-piece kids dinnerware" and "3-piece mens suit".

The term level features can be captured using the word vectors. To capture the contextual features of a term, we use the output from the *intent encoder* module. Consider the forward encoder GRU^f. At each position t, GRU^f updates the current summary $\mathbf{h}_{t-1}^f$ with the term v_{q_t} at point t. The contribution of the term v_{q_t} towards

defining the product intent of the query should be manifested in the extent to which it updates the summary at point $t-1$. Therefore, the contribution of a term at position t towards defining the product intent of the query should be a function of the difference in the summaries at the positions t and $t-1$, i.e., $(\mathbf{h}^f_t - \mathbf{h}^f_{t-1})$. Similarly, for the reverse encoder GRU^b, it should be a function of the difference in the summaries at the positions t and $t+1$, i.e., $(\mathbf{h}^b_t - \mathbf{h}^b_{t+1})$.

For each term q_t at position t in the query, the output of the feature generator (f_t) is the concatenation of the vectors $\mathbf{r}_{v_{q_t}}$, $\mathbf{h}^f_t - \mathbf{h}^f_{t-1}$ and $\mathbf{h}^b_t - \mathbf{h}^b_{t+1}$, i.e., $f_t = [\mathbf{r}_{v_{q_t}}, \mathbf{h}^f_t - \mathbf{h}^f_{t-1}, \mathbf{h}^b_t - \mathbf{h}^b_{t+1}]$.

Weight calculator: This module takes as input the features f_t generated by the feature generator for a term v_{q_t} and outputs a weight indicating the importance of the term towards defining the product intent of query q. We model the weight calculator as a multilayer perceptron (MLP) with a single node at the output layer. The *weight calculator* outputs a weight between 0 and 1. We refer to the weight corresponding to the query term v_{q_t} as $s(v_{q_t})$. To train the model, we minimize the binary cross entropy loss [7].

4 EXPERIMENTAL METHODOLOGY

Dataset: Our approach rely on training data, that contains query pairs of the form $(q, R(q))$ such that the user is unable to find the intended product from the results retrieved for the query q. Within the same session, he reformulates the query to $R(q)$, and is able to find the required products from the results displayed for $R(q)$. As discussed earlier, we obtain this training data by analyzing historical search logs. Since not all the reformulations satisfy our requirement, we follow a sequence of filtering steps so as to derive $(q, R(q))$ with the required properties. More details about the dataset and filtering steps are present in [5]. We restricted our training and evaluation to the rare queries, thus, constructed our dataset to only contain the reformulations where the initial query is a rare query.

Evaluation methodology and metrics: We evaluate our approach on two tasks: (i) improving retrieval of the relevant products, and (ii) predicting the query terms that are retained in the reformulation.

For the retrieval task, we measure the impact on retrieval by incorporating the estimated term weights in the BM25F-based ranking function, and measuring how well it improves the ranking of the relevant products. We incorporate the estimated term-weights in the ranking function by scaling (boosting) the individual BM25F score of each query term with the estimated term-weight, and using the modified scores to rank the products. We calculate the Mean Reciprocal Rank (MRR), which is the average of the reciprocal of the rank of the relevant product in response to a query. The relative performance improvement in MRR is given by $MRR_{ratio} = MRR_{boost}/MRR_{BM25F}$, where MRR_{boost} is the MRR when we have scaled the BM25F scores of each term with the corresponding term weight, and MRR_{BM25F} is the MRR corresponding to the BM25F scores of each term as it is.

For the second task, we looked at how well our approach is able to estimate higher weights for the terms that are retained in the reformulation (terms in the set $q \cap R(q)$), than the terms that are dropped (terms in the set $q \setminus R(q)$). We used the Precision@k ($P@k$) metric for evaluation. Given a query, $P@k$ measures what fraction of the top k terms, ranked by their estimated term-weights, are also present in the reformulation. We use $k = nnz$, where nnz is the number of total terms that were retained the reformulation, ignoring the stop-words. We report the averaged $P@nnz$ ($AP@nnz$) over all the test instances. We also present results for $k = 1, 2, 3$ as the queries in product search tend to be shorter.

Table 1: Results for the query term-weighting problem.

Model	MRR_{ratio}	$AP@nnz$	$AP@1$	$AP@2$	$AP@3$
CTW*	**1.030**	**0.746**	**0.792**	**0.804**	**0.832**
FTW	1.019	0.699	0.746	0.750	0.790
TF-IDF	1.006	0.548	0.514	0.603	0.700
VPCG & VG	0.694	0.558	0.537	0.616	0.709

* The performance was statistically significant (p-value < 0.05, paired t-test) against all competing methods.

Baselines: We compared CTW against two context-unaware approaches: VPCG & VG [3] and Frequentist Term-Weighting (FTW). VPCG & VG uses a regression model to fit a weighted linear combination of the representation of individual units (terms, n-grams) of queries/documents to that of the representations of the corresponding queries/documents. FTW estimates the weight of a term as the probability of the term present in a reformulated query, given that the query is also present in the initial query. Using the maximum likelihood estimation, the term-weight ($s(v_{q_t})$) of term v_{q_t} of the query q is calculated as,

$$s(v_{q_t}) = \frac{\sum_{q' \in Q} 1(v_{q_t} \in R(q'), v_{q_t} \in q')}{\sum_{q' \in Q} 1(v_{q_t} \in q')},$$

where $\sum_{q' \in Q} 1(v_{q_t} \in R(q'), v_{q_t} \in q')$ denotes the number of times the term v_{q_t} occurs in any query $q' \in Q$ and is retained in the reformulated query $R(q')$. $\sum_{q' \in Q} 1(v_{q_t} \in q')$ denotes the number of times v_{q_t} occurs in the initial query $q' \in Q$. Additionally, to illustrate that ranking terms according to their discriminating power is not suitable in the context of e-commerce, we also compared CTW against term frequency-inverse document frequency (TF-IDF).

5 RESULTS AND DISCUSSION

Quantitative evaluation: Table 1 shows the performance statistics for the term-weighting problem for the various approaches. The intent term weighting performed by our proposed approach CTW, FTW and TF-IDF leads to better ranking of the search results, as shown by the improvement on the MRR metric. CTW leads to the maximum increase in the MRR, achieving an improvement of 3% against the case when no intent term weighting is applied. This illustrates the advantage of using contextual knowledge to filter out important terms from a search query. Moreover, the poor performance of the TF-IDF as compared to the non contextual baseline (FTW) shows that the discriminatory power of a term does not necessarily correlate with the importance of that term towards defining the product intent of that query. Our competing approach VPCG & VG gives worse ranking of the search results as compared to the baseline BM25F based retrieval algorithm, further demonstrating the advantage of using contextual knowledge to filter out important terms from a search query, especially for the tail queries. Figure 2 shows how the MRR is affected with the increase in the query length. For all query lengths, CTW gives a better ranking of the

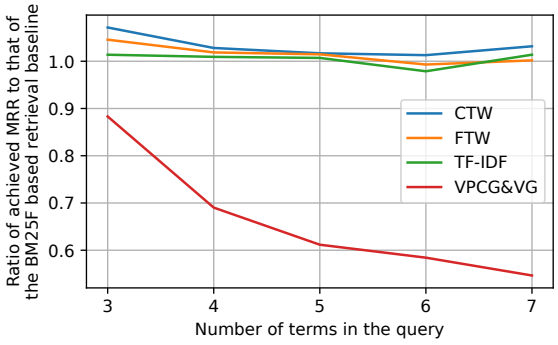

Figure 2: Variation of the MRR with query length.

Table 2: Predicted term-weights for a some queries. The bold terms refer to the terms retained in the reformulated query.

(a) Initial query is "battery night light with timer" and the reformulated query is "munchkin night light".

Model	battery	**night**	**light**	with	timer
CTW	0.73	0.80	1.00	0.51	0.29
FTW	1.00	0.77	0.88	0.49	0.97
TF-IDF	0.62	0.80	0.61	0.48	1.00
VPCG & VG	0.83	0.09	1.00	0.00	0.53

(b) Initial query is "work boots steel toe breakers" and the reformulated query is "survivor work boots mens".

Model	**work**	**boots**	steel	toe	breakers
CTW	0.74	1.00	0.85	0.79	0.41
FTW	0.51	0.76	0.65	0.72	1.00
TF-IDF	0.62	0.51	0.58	0.69	1.00
VPCG & VG	0.01	1.00	0.56	0.01	0.12

(c) Initial query is "auto seat cover wonder woman" and the reformulated query is "auto seat cover".

Model	**auto**	**seat**	**cover**	wonder	woman
CTW	0.78	1.00	0.94	0.39	0.21
FTW	0.52	0.85	0.59	0.91	1.00
TF-IDF	0.92	0.71	0.74	1.00	0.73
VPCG & VG	0.19	0.10	0.07	1.00	0.16

search results than the baselines. However, for all the approaches, the performance usually drops with the increase in query length, as noisy terms are expected to appear in longer queries.

The same trend is demonstrated by the other metrics too. The relative performance gain of the CTW over FTW, in terms of the *AP@nnz* is 6.7%. Even on *AP*@1, *AP*@2 and *AP*@3, CTW performs significantly better than the FTW. We provide further qualitative discussion about this in the following section.

Qualitative evaluation: In order to visualize the weights produced by the CTW and compare them with the non-contextual baselines, we looked into a few queries and the weights estimated for the terms in them by various approaches. Table 2 shows the predictions for some of the queries. The weights are normalized so that the maximum weight assigned to a term is one. Table 2a shows an example of a query where the user reformulated it to capture the generalized product intent. The initial query is "battery night light with timer", for which the product intent is a *night light* with some particular attributes. The product intent for the query was not satisfied, and the user reformulated the query to "munchkin night light". Without the contextual information, it would be difficult to estimate the product intent because of the presence of the terms like *battery* and *timer*. TF-IDF gives highest weight to the term *timer*, because *timer* is less frequent as compared to the other terms, but the term *timer* will produce irrelevant results if it is not accompanied with the *night light*. Similarly, FTW gives highest weight to the term *battery* because *battery* is a common term which defines the intent for many queries like "battery operated fan", "battery charger", etc. The term *battery* is retained in a large number of reformulations, thus FTW gives it high weight. In a similar manner, VPCG & VG gives higher weight to the term *battery* than the term *night* because *battery* is a common term which defines the intent for many queries. In comparison, CTW is able to estimate the correct product type and gives highest weights to the terms *night* and *light*. Similar trend is shown by the other examples.

6 CONCLUSION

Our work showed that query reformulation could be used as a form of distant supervision to determine the product intent of a query. The product intent of a term in the query depends on the context of the term, i.e., neighboring terms, in the query and developing context-aware methods lead to better performance over non-contextual baselines.

ACKNOWLEDGMENTS

This work was supported in part by NSF (1447788, 1704074, 1757916, 1834251) and Walmart Labs. Access to research and computing facilities was provided by the Minnesota Supercomputing Institute.

REFERENCES

[1] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).
[2] Doug Downey, Susan Dumais, and Eric Horvitz. 2007. Heads and tails: studies of web search with common and rare queries. In *SIGIR*.
[3] Shan Jiang, Yuening Hu, Changsung Kang, Tim Daly Jr, Dawei Yin, Yi Chang, and Chengxiang Zhai. 2016. Learning query and document relevance from a web-scale click graph. In *SIGIR*.
[4] Shubhra Kanti Karmaker Santu, Parikshit Sondhi, and ChengXiang Zhai. 2017. On application of learning to rank for e-commerce search. In *SIGIR*. ACM.
[5] Saurav Manchanda, Mohit Sharma, and George Karypis. 2019. Intent term selection and refinement in e-commerce queries. *arXiv preprint arXiv:1908.08564* (2019).
[6] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL-IJCNLP*.
[7] Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. 2014. Large-scale multi-label text classification-revisiting neural networks. In *ECML PKDD*.
[8] Stephen E Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR*.
[9] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at TREC-3. *Nist Special Publication Sp* (1995).
[10] Gerard Salton and Michael J McGill. 1986. Introduction to modern information retrieval. (1986).
[11] Yangqiu Song, Haixun Wang, Weizhu Chen, and Shusen Wang. 2014. Transfer understanding from head queries to tail queries. In *CIKM*. ACM.
[12] Ellen M Voorhees et al. 1999. The TREC-8 Question Answering Track Report.. In *Trec*, Vol. 99. 77–82.