A Dynamic Product-aware Learning Model for E-commerce Query Intent Understanding

Jiashu Zhao
Department of Physics and Computer Science,
Wilfrid Laurier University
Waterloo, Canada
jzhao@wlu.ca

Hongshen Chen, Dawei Yin*
Data Science Lab,
JD.COM
Beijing, China
ac@chenhongshen.com,yindawei@acm.org

ABSTRACT

Query intent understanding is a fundamental and essential task in searching, which promotes personalized retrieval results and users' satisfaction. In E-commerce, query understanding is particularly referring to bridging the gap between query representations and product representations. In this paper, we aim to map the queries into the predefined tens of thousands of fine-grained categories extracted from the product descriptions. The problem is very challenging in several aspects. First, a query may be related to multiple categories and to identify all the best matching categories could eventually drive the search engine for high recall and diversity. Second, the same query may have dynamic intents under various scenarios and there is a need to distinguish the differences to promote accurate categories of products. Third, the tail queries are particularly difficult for understanding due to noise and lack of customer feedback information. To better understand the queries, we firstly conduct analysis on the search queries and behaviors in the E-commerce domain and identified the uniqueness of our problem (e.g. longer sessions). Then we propose a *D*ynamic *P*roduct-aware *H*ierarchical *A*ttention (*DPHA*) framework to capture the explicit and implied meanings of a query given its context information in the session. Specifically, *DPHA* automatically learns the bidirectional query-level and self-attentional session-level representations which can capture both complex long range dependencies and structural information. Extensive experimental results on a real E-commerce query data set demonstrate the effectiveness of the proposed *DPHA* compared to the state-of-art baselines.

CCS CONCEPTS

• **Information systems → Information retrieval**.

KEYWORDS

Query intent, Attention, Product-aware, Hierarchical neural network

*Corresponding Author

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

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3358055

ACM Reference Format:
Jiashu Zhao and Hongshen Chen, Dawei Yin. 2019. A Dynamic Product-aware Learning Model for E-commerce Query Intent Understanding. In *Proceedings of The 28th ACM International Conference on Information and Knowledge Management (CIKM '19).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3357384.3358055

1 INTRODUCTION

Accurate Query understanding is crucial for better retrieval results in various search scenarios. In particular, the major e-commerce sites usually include billions of products, so customers' shopping experience could be improved by finding the best matching categorizes, and eventually the overall revenue could be increased. For example, the query could be a function that the customer would like the product to have, for example "moisture proof", while this query does not directly include a product name. By finding the the related product categories from the inventory that can achieve the function, such as "Dehumidifiers", "Drying box" and etc., the search system can then return the right products for customers to choose. In this paper, we aim to identify the customers' dynamic query intent with respect to the product categories. For E-commerce, since the product inventory is well-maintained by a large amount of resources, including human labelling a new product when it is newly added, as well as the automatic tagging models learned from the products' rich textual information of detailed descriptions, the product categories have high qualities and fine-grained. Here the fine-grained product categories are the most determinative and specific words to describe the product, rather than general categories. We can start from the product categories and map queries the product category representations to bridge the gap between queries and products.

Since queries are often short and vague [19], the user behaviors which are recognized to include rich information about users' interests and preferences, have been widely used for query understanding [4, 7, 8, 20]. However, there has not been through study on the characteristics of E-commerce queries and how to link the queries with the products that the customers would like to purchase. The query understanding is not a trivial task, and we summarize the main challenges in finding the best matching product categories for a query as follows. (1) Non-Exclusiveness: A query may have more than one correct product categories because of multiple intents of the query, and synonymous, inclusions and overlaps of product categories. For example, a query "Samsung" represents a brand and could mean multiple specific electronics, such as "Cell Phone", "Headset", "Storage Card", etc.. The categories "Bath Pillow" and "Spa Cushion" are synonymous which could be complement to each other. It is also possible that one category includes the meaning of

another, such as "Rice Container" is one of "Kitchen Organizer". (2) Dynamic Intents: The intent of the same query may not be the same with different customers, or even with the same customer under different circumstances. For instance, "Apple" could mean a brand with several categories of electronic products, or a type of fruit. (3) Lack of User Behaviors for tail queries: For the large amount of long tail queries, there might not be any click information to learn from.

Inspired by the above challenges, we propose a *D*ynamic *P*roduct-aware *H*ierarchical *A*ttention (*DPHA*) framework for learning the predictive structure from the session information in addition the the query's text, which includes all the preceding queries and user behaviors within a short period of time. The proposed framework has three major modules. We first build a query-based module, which has a bidirectional (forward and backward) Recurrent Neural Networks (RNN) with Gated Recurrent Unit (GRU) to learn the encoding of the words in the query. Then an attention mechanism generates dynamic query representations from the word encoders. The second module is at the session level, which uses self attention and GRU dropout techniques to learn the dependencies between the queries and to avoid overfitting. The last step takes the session annotations and the query annotations as inputs and predicts the product categories with a regularization from the next query.

The main contributions of this work are summarized as follows.

- We thoroughly analyze query distributions and characteristics for product search in E-commerce. Most previous query analysis are conducted for Web search, and this paper brings new perspectives to the query understanding in the specific product search domain.
- We formally define the query understanding problem in product search as mapping queries into product categories with the dynamic user information needs.
- To address the above defined problem with the analyzed query characteristics, we design a *D*ynamic *P*roduct-aware *H*ierarchical *A*ttention (*DPHA*) framework with regularization from the subsequent category which could predict the dynamic query intent in product search.
- The experiments are conducted on a real large data set from an E-commerce site, and the results have illustrated the effectiveness of the proposed framework.

2 RELATED WORK

There are multiple research directions in query intent understanding, including to recognize the query's type [1] (e.g. commercial or noncommercial, navigational or informational), to identify the intent words in the query [28], or classify queries into target categories [29], to recognize the named entities and [13] temporal intent [14]. For search engines in E-commerce sites, our primary goal is to understand which types of products the customer is looking for. Therefore, we fix a large set of categorizes generated from the products and then classify the queries accordingly to bridge the gap between queries and products.

Since the queries are short, the external information are often incorporated to enrich the query information, including Wikipedia [18], external probabilistic knowledge base [37], retrieved documents of the queries [6] and query logs [1]. As a vertical search in the area of E-commerce, it is not practical to use any external information outside of the site. All the retrieved products should be in the site's stock. Otherwise, even if a query is understood correctly but it can not be linked to any of the stocking products, it will still be recognized as a bad case. In this paper, we adopt the search logs with customers' behaviors on this site and the corresponding clicked the product categories for better modelling the query intent.

The query classification technologies include word matching [5], unsupervised approaches such as topic modelling [3] and building concept graphs [11], or conventional classification approaches such as Naive Bayes classifier [39], Gradient Boosting Trees, Support Vector Machine, Random Forest with manually crafted features [22], sequential learning model, such as Conditional Random Field (CRF) [7]. In recent years, the popular neural models are emerging in various query applications [17, 23, 25, 31], including but not limited to query suggestion, reformulation, expansion and intent understanding. Convolutional Neural Networks (CNN) has been adopted to extract query vector representations as the features for Random Forest to classify queries in [15]. A deep neural network is designed for a multi-task retrieval problem including identifying the query's search domain as well as document ranking in [24]. By stacking multiple LSTM units to learn query representations and using CNN for query classification, the deep model has obtained higher performance compared to conventional classification approaches and variations of CNN models in [30]. The Global Vectors model enriched word embedding vectors with multiple Bidirectional LSTM layers is proposed in [32] to understand query intent. In this paper, we designed a hierarchical deep learning framework which learns the query representations, session representations and predicts the product categories that can best describe the query's intent dynamically.

3 PROBLEM DEFINITION AND ANALYSIS

3.1 Preliminaries

For a given query $\tilde{q}_i$, the intent of the query might be different under various scenarios, which could be captured by the preceding queries and clicked products in the same session. Since multiple users might search for the same query, and one user may search the same query for multiple times, a search engine could observe a set of sessions for $\tilde{q}_i$: $\tilde{S}_i = \{\tilde{S}_{i,k}\}_{k=1}^{K_i}$, where K_i is the total number of sessions that include $\tilde{q}_i$. For each session $\tilde{S}_{i,k}$, it includes a sequence of queries and corresponding user behaviors for the same user on the same device within a short period of time. The consecutive user behaviors in a search session are often recognized to be likely to reflect similar or correlated search intent. Multiple approaches could be utilized for session segmentation, and we adopted a commonly used approach, which separates the sessions if the user has not issued any query in 30 minutes [16].

A session $\tilde{S}_{i,k}$ for the query $\tilde{q}_i$ is represented as $\{(\tilde{q}_{i,k,j}, \tilde{p}_{i,k,j})\}_{j=1}^{J}$ where J is the length of the session, $\tilde{q}_{i,k,j}$ is a query in the session and $\tilde{p}_{i,k,j}$ is the clicked product correspondingly. Here we have not included the queries without any clicked product, since they are less informative and more likely to bring noise to the learning process. In this paper, we focus on identifying the interested product category of a search query. Due to the large size of products

No.	Query	Preceding Session: (Query, Clicked Category)	Intent
1	Watch	(Watch, Quartz Watch), (Watch, Watch), (Watch, Quartz Watch)	Quartz Watch, Watch
2	Watch	(Watch, Men's Watch), (The horse, Men's Watch),(Watch, Quartz Watch), (Watch, Men's Watch)	Men's Watch, Watch
3	Watch	(Watch, Mechanical watch), (Mechanical Watch, Mechanical watch)	Mechanical watch, Watch
4	Apple	(Orange, Ugly Orange), (Durian, Fruit), (Children's leggings, Children's underwear), (Children's leggings, Children's pajamas)	Apple, Fruit
5	Apple	(Google, Cell Phone), (Samsung, Cell Phone)	Cell Phone
6	Apple	(Ipad, Tablet), (Surface, Tablet)	Tablet

Table 1: Examples of Queries and the Preceding sessions

(usually in multiple millions to even billions), we directly represent the sessions by product categories to avoid sparseness. Assume that we have category information for all the products in stock, we use product category to represent the product item for simplicity and convenience. If there are multiple fine-grained categories for one product, only the best matching category is selected. For each query issued in the session $\tilde{S}_{i,k}$, $(\tilde{q}_{i,k,j}, \tilde{p}_{i,k,j})$ is composed of query words in $\tilde{q}_{i,k,j}$ and the category words in $\tilde{p}_{i,k,j}$.

The goal of our query understanding problem is defined as to predict the query intent at the time of the customer issuing the query $\tilde{q}_i$, given the session information before $\tilde{q}_i$, which is $\{(\tilde{q}_{i,k,j}, \tilde{p}_{i,k,j})\}_{j=1}^{j_c}$, where j_c is the last position before the occurrence of $\tilde{q}_i$ in the session. For this sequential predictive problem, only the user behaviours before query $\tilde{q}_i$ are available for the system to access in real application and in the design of our test data set. Table 1 shows examples of two queries "Watch" and "Apple" and several short sessions for the queries. The query "Watch" itself is a type of products, while the customers' interest could be more focused as "Quartz watch" or "Mechanical watch". A vague query "Apple" could be a brand or a category of fruits. Even with "Apple" as a brand only, the related product categories could be "Cell phone", "Tablet", "Laptop", "Desktop" and etc.. the sessional information could bring additional understanding of the query's real search intent. We can see that query q_i may have appeared in the session previously or firstly issued in the session. The user may switches a couple of topics in the middle of a session as well. For example in Line Number 4 of Table 1, the customer looks for some fruits and then children's' clothes, and then switches back to fruits.

In the training phrase, the entire session $\tilde{S}_{i,k} = \{(\tilde{q}_{i,k,j}, \tilde{p}_{i,k,j})\}_{j=1}^{J}$ could be obtained. The clicked category of the present query $\tilde{q}_i$ is used as label, and the subsequent behaviors in the session could serve as additional information to model. The examples in Table 1 only include short and relatively clean sessions. In practice, the sessions could be informative as well as noisy. In order make accurate predictions, a model needs to be able to capture the useful information from the query and the session as well as to eliminate the irrelevant impacts.

3.2 Query Analysis

We collected 450,294,680 queries from an E-commerce site over 6 months in 2018 to study the distributions of the queries and the sessions.

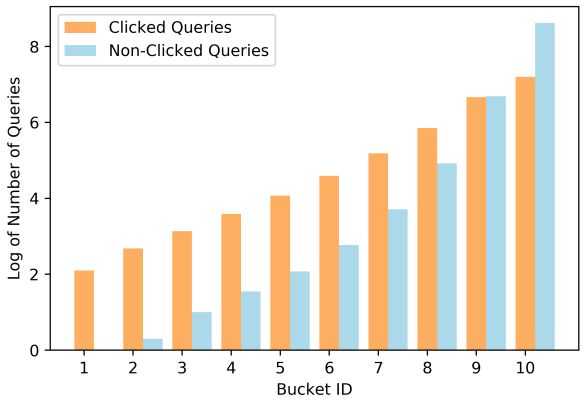


Figure 1: Distribution of Queries with Bucket ID and Clicks

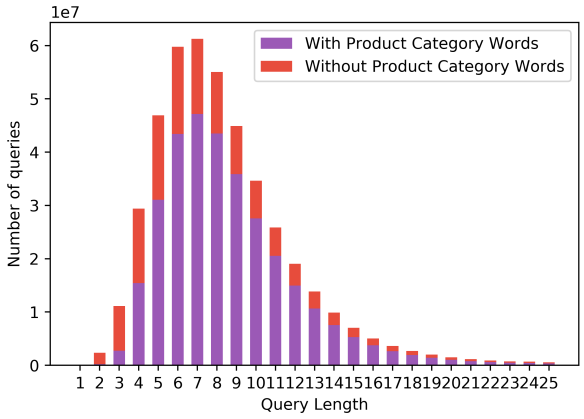


Figure 2: Query Distribution over Length

3.2.1 Search traffics and Clicks of Queries. The search traffic (frequency) for a query is the one of the most crucial characteristics of a query, and many sites employ distinctive search strategies for queries with different search traffics. We firstly splits the queries into 10 buckets according to the search frequencies, where each bucket has approximate the same amount of exposures. A smaller bucket number indicates more frequent queries. Since the range of the numbers in the buckets is very large, we plot the log of these numbers in Figure 1. Different colors in the bar chart indicate

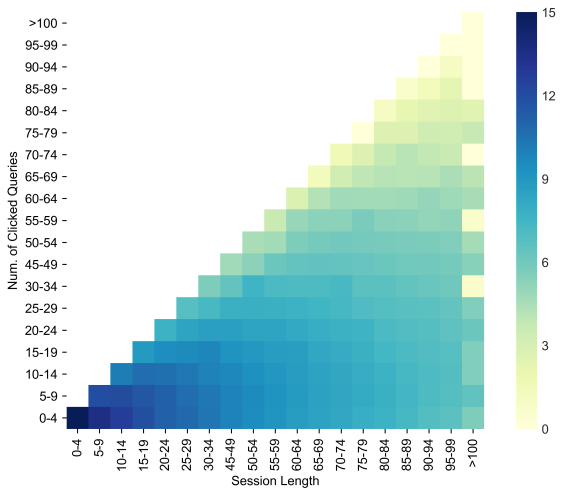


Figure 3: Heatmap of Session Length and Number of Clicked Queries (The values are transformed by the log function.)

whether the queries are clicked or not during 6 months period. We can see that most top and torso queries are clicked, while the tail queries are rarely clicked. There is no direct user behavior information for the unclicked queries, and thus they are more difficult to classify.

3.2.2 Lengths and Category Words of Queries. We further analyze the query length and product category words distribution in the queries. Overall, the query length ranges from 1 to around 100, and the average is around 8-9 characters. The details are shown in Figure 2. Since the longer queries are rare, we only plotted queries with a length of less than 25. We can see that queries with a length of 7 have the largest amount and the amount gradually decreases for longer and shorter queries. Moreover, some queries include the product category words explicitly while the others not. In total, 71% of queries have product category words. In the figure, shorter queries (with a length around 3,4) tend to have a lower rate to include the category words.

3.2.3 Analysis of Query Sessions. A session is a sequence of consecutive queries with user behaviors. We follow the common practice in Web search log analysis by using 30-minute timeout for session segmentation [16]. We analyzed all the 180,730,032 sessions in 7 consecutive days. The average session length is 9.82, and the averaged number of clicked queries per session is 3.21. The session length is much long than reported in web search [19, 27]. In E-commerce, customers may spend longer time on browsing and comparing products. Figure 3 illustrates the details for the session length and the number of clicked queries. We grouped the number of sessions and clicked queries into number of 5 intervals to show the aggregated results. The numbers are transformed by a log function because of the large range. Even though many of the sessions are short, we can see that the customer may be active for very long sessions (around 100 queries in the sessions). We have very rich session context information to study user behaviors for various applications in E-commerce, especially for query understanding in this paper.

4 METHODOLOGY

In this section, we propose a *D*ynamic *P*roduct-aware *H*ierarchical *A*ttention (*DPHA*) framework and discuss the details of its components.

4.1 A Product-Aware hierarchical Framework

The overall framework is shown in Figure 4. We aim to capture the sessional information including the preceding queries and clicked product categories. The framework is designed according to the following characteristics of query intents. The query intent is dynamic across sessions, and different preceding queries have different influences on the present query's intent. In a finer granularity, the words in a query would contribute to the query's semantic meaning unequally. The basic idea of the proposed framework is to jointly learn a representation for each query and its preceding user behaviors in the same session. In particular, we first encode words of each query and generate the query's representation with word-level attention. And then the session-level attention is used to extract the most important information in all preceding queries. Finally, a hybrid representation of a query and its session is then adopted for model inference, where a future factor is incorporated as regularization.

4.2 Attentional Query Intent

At the query level, we treat the present query $\tilde{q}_i$ and the preceding queries in the session differently, since we do not have user behavior information for the present query at the time of prediction. For preceding queries, its words and the the corresponding clicked category are informative to understand the semantic intent of the query. Therefore, both query words and the category words are embedded from one-hot representations. Here we use RNN to learn the words' dense representations to model the sequence in the words. The long term dependencies among the words could be captured by Long short-term memory (LSTM)[34] and Gated Recurrent Unit (GRU) [9]. Here we adopt GRU as a basic RNN cell since it has a relatively simpler structure which has less computational complexity [12]. The activation of GRU is a linear interpolation between the previous activation and the candidate activation [10]. It calculates a series of latent state vectors according to the inputs and previous state vectors. The single layer GRU computes the hidden states as follows.

$$\mathbf{r}_{i,k,j,m} = \sigma(\mathbf{W}_r \mathbf{x}_{i,k,j,m} + \mathbf{U}_r \mathbf{h}_{i,k,j,m-1})$$
$$\mathbf{z}_{i,k,j,m} = \sigma(\mathbf{W}_z \mathbf{x}_{i,k,j,m} + \mathbf{U}_z \mathbf{h}_{i,k,j,m-1})$$
$$\mathbf{h}_{i,k,j,m} = (1 - \mathbf{z}_{i,k,j,m})\mathbf{h}_{i,k,j,m-1} + \mathbf{z}_{i,k,j,m}\hat{\mathbf{h}}_{i,k,j,m}$$
$$\hat{\mathbf{h}}_{i,k,j,m} = \tanh(\mathbf{W}\mathbf{x}_{i,k,j,m} + \mathbf{U}(\mathbf{r}_{i,k,j,m} \odot \mathbf{h}_{i,k,j,m-1})$$

where $\mathbf{h}_{i,k,j,m}$ is the new hidden state, $\mathbf{r}_{i,k,j,m}$ is the reset gate to decide how much of the past information to forget, $\mathbf{z}_{i,k,j,m}$ is the update gate to decide how much $\mathbf{h}_{i,k,j,m}$ updates, $\sigma(\cdot)$ is the sigmoid function, $\mathbf{W}_r$, $\mathbf{U}_r$, $\mathbf{W}_z$, $\mathbf{U}_z$, $\mathbf{W}$, and $\mathbf{U}$ are parameters. For a preceding query, $\mathbf{x}_{i,k,j,m}$ is an input from the query words and product category words in $(\tilde{q}_{i,k,j}, \tilde{p}_{i,k,j})$. For the present query, $\mathbf{x}_{i,k,j,m}$ is an input from the query words in $\tilde{q}_i$ only, since the

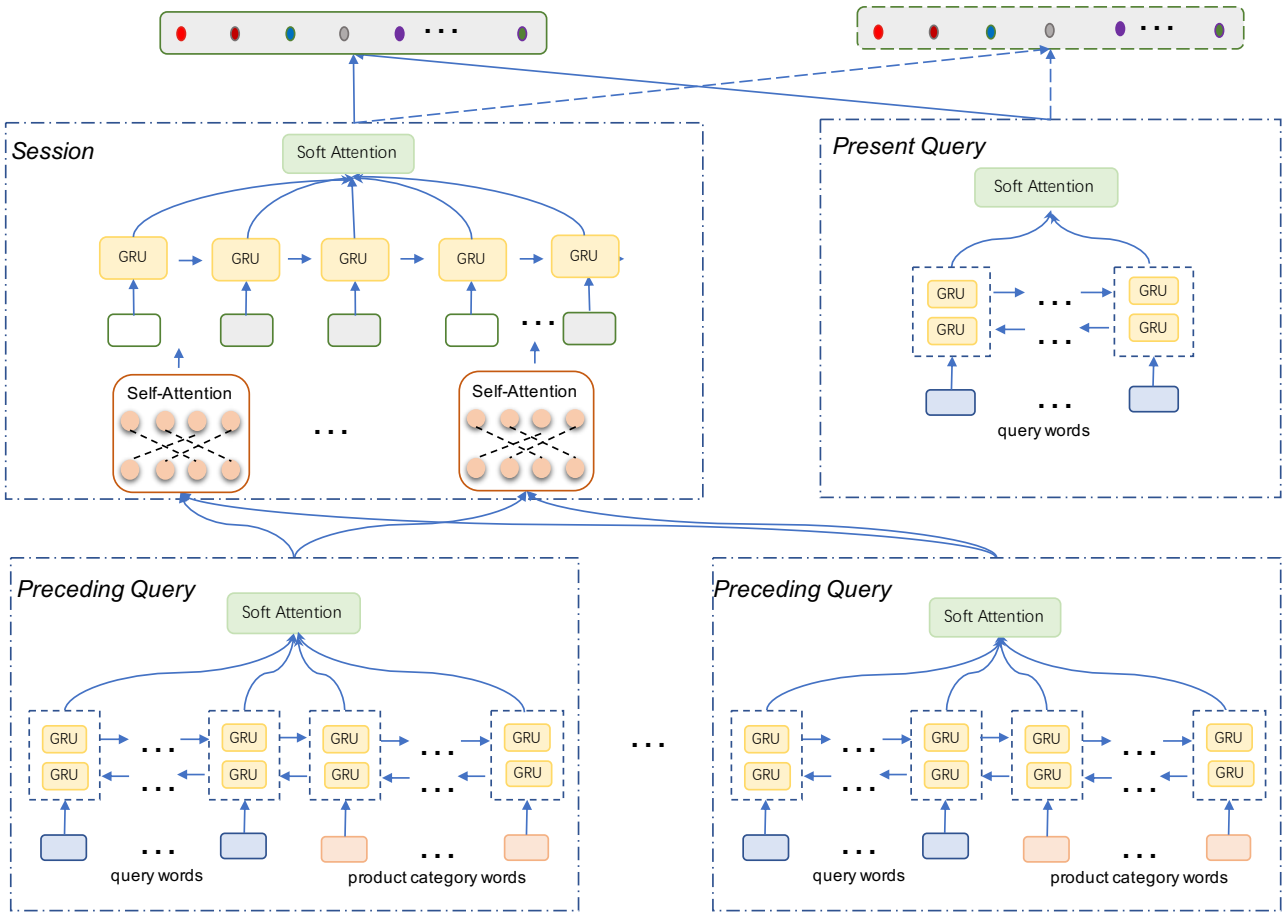


Figure 4: The Proposed *DPHA* Framework

clicked product category information is for prediction. To learn the full context information of the input query words and product category words, we exploit bidirectional GRU which generates forward hidden states $\overrightarrow{\mathbf{h}}_{i,k,j,m}$ and backward hidden states $\overleftarrow{\mathbf{h}}_{i,k,j,m}$. The concatenated annotation $Concat(\overrightarrow{\mathbf{h}}_{i,k,j,m}, \overleftarrow{\mathbf{h}}_{i,k,j,m})$ represents the summarized information of the whole query at the word $x_{i,k,j,m}$.

The words in the query and product category contribute differently to the query understanding task. Then after we obtained the word encoders from bidirectional GRU, the soft attention mechanism [38] is introduced here to generate the query's annotation by aggregating the words' annotations unevenly.

$$\mathbf{u}_{i,k,j,m} = \tanh(\mathbf{W}_w \mathbf{h}_{i,k,j,m} + \mathbf{b}_w)$$
$$\alpha_{i,k,j,m} = softmax(\mathbf{u}_{i,k,j,m}\mathbf{u}_w)$$
$$\mathbf{q}_{i,k,j} = \Sigma_m \alpha_{i,k,j,m}\mathbf{u}_{i,k,j,m}$$

where $\mathbf{u}_{i,k,j,m}$ is a hidden representation of the input vector $\mathbf{h}_{i,k,j,m}$, then $\mathbf{u}_w$ is a parameter vector which learns the importance of of the vector space. Here $\alpha_{i,k,j,m}$ yields a probabilistic interpretation of Attention. Finally, a query vector $\mathbf{q}_{i,k,j}$ is the expectation of the important words. Here we have different inputs for preceding queries and the present query into the attention mechanism from the outputs of the bidirectional GRU. For the present query, its query vector represents the weighed sum of the query words only. And for the preceding queries, its query vector represents the weighted sum of query words and clicked product category words annotations. In the next step, we model the dynamic session intent from the preceding queries.

4.3 Learning Dynamic Self-Attention Session Intent

Within a session, the customer may reformulate the query in various forms to serve the same intent, or even search the same query multiple times and clicked different (categories of) products. Then we apply multi-head self-attention mechanism [35, 36] to learn from the session itself. The single scaled dot-product self-attention function is as follows

$$M_n(\hat{\mathbf{S}}_{i,k}) = softmax(\mathbf{Q}\mathbf{W}_n^Q \mathbf{K}(\mathbf{W}_n^K)^T/\sqrt{j_c})\mathbf{V}\mathbf{W}_n^V$$

where $\hat{\mathbf{S}}_{i,k} = [\mathbf{q}_{i,k,1}; ...; \mathbf{q}_{i,k,j_c}]$ is the sessional input with the attentional query annotations, j_c is the length of the preceding

session, $\mathbf{W}_n^Q$, $\mathbf{W}_n^K$, and $\mathbf{W}_n^V$ are parameters. Multiple linear projections with parallel heads are employed to jointly capture the focus on different parts of session sequence. The vectors from multiple heads are concatenated and linearly transformed as the output of the self-attention layer.

$$MultiHead(\hat{\mathbf{S}}_{i,k}) = Concat(M_1(\hat{\mathbf{S}}_{i,k}), ..., M_{nh}(\hat{\mathbf{S}}_{i,k}))\mathbf{W}^O$$

where nh is the total number of attention layers, and $\mathbf{W}^O$ is the parameter. To capture the position in the sessional sequence, positional encodings are added to the input separately for odd and even dimensions by

$$\begin{cases} PE_{(j,2l+1)} = sin(j/10000^{2l/d_e}) \\ PE_{(j,2l)} = cos(j/10000^{2l/d_e}) \end{cases}$$

where j is the query's position in the session, l is the dimension and d_e is the total dimension of the embedding. The session's embedding representation is generated with the ability of capturing the influence between any queries in the session by ignoring the distances. This is particularly useful for the dynamic session intent understanding. The customer's intent may shift overtime and back-and-forth, for example, in the 4th line of Table 1, the customer was initially searching for some fruits, then thought about children's clothes, and came back to the original intention of buying fruit again. The preceding queries in the session that are farther have quite significantly influential to the present query.

Moreover, the shifting of customers' intention in the same session might bring noise to the present query understanding. Some totally random queries may come up, and the customer may input a typo or searched an unintended query by mistake. In this case, we incorporated the dropout [33] method, which randomly samples units for the next layer. The query units are assumed to present in the session with a certain probability in the training process.

Similar to the query-level, we then adopt the GRU to encode the session by learning the sequential information in the session forwardly from $q_{i,k,1}$ to q_{i,k,j_c}.

$$\begin{aligned} \mathbf{r}_{i,k,j} &= \sigma(\mathbf{W}'_r\mathbf{q}'_{i,k,j} + \mathbf{U}'_r\mathbf{h}_{i,k,j-1}) \\ \mathbf{z}_{i,k,j} &= \sigma(\mathbf{W}'_z\mathbf{q}'_{i,k,j} + \mathbf{U}'_z\mathbf{h}_{i,k,j-1}) \\ \mathbf{h}_{i,k,j} &= (1 - \mathbf{z}_{i,k,j})\,\mathbf{h}_{i,k,j-1} + \mathbf{z}_{i,k,j}\hat{\mathbf{h}}_{i,k,j} \\ \hat{\mathbf{h}}_{i,k,j} &= \tanh(\mathbf{W}'\,\mathbf{q}'_{i,k,j} + \mathbf{U}'(\mathbf{r}_{i,k,j} \odot \mathbf{h}_{i,k,j-1}) \end{aligned}$$

where $\mathbf{r}_{i,k,j}$ and $\mathbf{z}_{i,k,j}$ are the reset and update gates, the input $q'_{i,k,j}$ is the output from the previous layer, $\mathbf{h}_{i,k,j}$ is the new hidden state, $\mathbf{W}'_r$, $\mathbf{U}'_r$, $\mathbf{W}'_z$, $\mathbf{U}'_z$, $\mathbf{W}'$, and $\mathbf{U}'$ are parameters for the session-level GRU. And then another soft attention layer is applied to generate the session's annotation $\mathbf{s}_{i,k}$ by aggregating the hidden states.

$$\begin{aligned} \mathbf{u}_{i,k,j} &= \tanh(\mathbf{W}'_w\mathbf{h}_{i,k,j} + \mathbf{b}'_w) \\ \alpha_{i,k,j} &= softmax(\mathbf{u}_{i,k,j}\mathbf{u}'_w)) \\ \mathbf{s}_{i,k} &= \Sigma_j\alpha_{i,k,j}\mathbf{u}_{i,k,j} \end{aligned}$$

4.4 Model Inference and Prediction

The present query $\tilde{q}_i$'s intent is predicted based on $\tilde{q}_i$'s textual words as as well as the sessional information. The representations from the sessional attention layer and present query attention layer are features for category prediction. Then a softmax function outputs the probabilities for categories.

$$P_c(\mathbf{v}) = softmax(\mathbf{W}_c\mathbf{v} + \mathbf{b}_c)$$

where the input v is the concatenation of the session and query context vectors $v = Concat(\mathbf{S}_{i,k}|\mathbf{q}_i)$, $\mathbf{W}_c$ and $\mathbf{b}_c$ are parameters for the scoring function.

There could be circumstances for a query not having enough information from the preceding session, for example, a query issued as an early stage of the session did not return satisfactory results and then the customer rewrite the query with more clear intent. In such cases in the training data, the insight from the future information in the session could also be very helpful. However, we should be careful when adopting such context since it won't be available in the real-time system or the testing data. In this paper, we design a second prediction in the model which serves the purpose of regularization from the future information. First, it uses a softmax function to predict the multiclass classification separately.

$$P_f(\mathbf{v}) = softmax(\mathbf{W}_f\mathbf{v} + \mathbf{b}_f)$$

Then we utilize the combined cross entropy as the metric in our loss function which is defined as follows:

$$L = -\Sigma_v(y_c log(P_c(\mathbf{v})) - \lambda_v * \Sigma_v(y_f log(P_f(\mathbf{v}))$$

where y_c and y_f are the clicked categories for the present query and the subsequent query, and λ_v is the parameter. Here the second part in the loss function from the subsequent query passes the the information in the future to the parameter training process. In prediction, the framework outputs both $P_c(\mathbf{v})$ and $P_f(\mathbf{v})$. The predicted present category $P_c(\mathbf{v})$ is used in evaluation, while $P_f(\mathbf{v})$ will be neglected.

5 EXPERIMENTAL SETTINGS

In this section, we describe the experimental settings including data sets, parameter settings, evaluations metrics and baselines for comparison.

5.1 Data Sets and Parameter Settings

We perform extensive experiments with real query log data sets from a real-world e-commerce site[1]. When a customer accesses the site, a query is issued and then the relevant product maybe clicked. The logged data sets includes the entire sequence of queries and the clicked/non-clicked product information. We collected session data for 14 days in March 2019, where the training data set is obtained from the first 7 days, and we sampled the testing set from the last 7 days of thee data. Since the manual annotations of query's intention is expensive to get for a large amount of data, the user clicked product's category is used as label for training. The training set is automatically generated with 19,034,352 randomly sampled entries, each of which includes a preceding session, a present query, a present product category, and a subsequent product category. The product category set is obtained from the e-commerce site, which includes 61,773 categories to predict.

For the testing set, we employed stratified sampling [2] based on the queries' search engine traffic in order to fairly evaluate

[1] http://www.jd.com

the models' generalized performance. We randomly sampled 200 queries from 10 traffic buckets, and each query include up to 10 sessions in the testing data. For the less frequencies in bucket 8,9 and 10, there are less number of sessions available. So the testing entries in these three buckets are merged. Different from the training set, the queries without clicks are included in the testing set, since the framework may handle all possible queries in real application.In total, the testing data set includes 15,011 entries, and the generated results are manually annotated by human experts.

In the proposed framework, the embedding dimensions are all set to 100 throughout the paper. The maximum number of words in each query is set to 20, and the maximum queries in each session is set to 10. We employ 8 heads in the self attention layer. The RMSprop optimization [26] is adopted in training. 30% of the training data is used for validation purpose. The proposed framework predicates a score for each product category and the top ranked categories are returned as the model output.

5.2 Evaluation Metrics

In this paper, we use the following metrics to evaluate the effectiveness of query categorization .

- *Precision* is the fraction of identified categories that are real intent of the query.
- *Recall* is the fraction of the correct categories that are successfully identified.
- $F-Score$ is the harmonic mean of precision and recall, which is a single metric to evaluate a model in terms of a balance between the precision and recall: $\frac{2*Precision*Recall}{Precision+Recall}$.

Since we are more concerned with the top identified categories among all the tens of thousands of categories, we evaluate the proposed framework and the baselines on the above evaluation metrics for the topmost 3 results, i.e. $Precision@1$, $Precision@2$, $Precision@3$, $Recall@1$, $Recall@2$, $Recall@3$, $FScore@1$, $FScore@2$, and $FScore@3$.

5.3 Comparison Baselines

We compare the proposed framework to several sate-of-the-art models in the area of query-documentation representation and multiclass classification. The first approach for comparison is a simple *LiteralMatching*, which is fast and easy to explain. The second group of approaches are from [21], denoted as *ClickGraph* and $ClickGraph-VG$ here, are based on the query clicked document graph. Then for the deep learning baselines, we choose $BiLSTM-Q$, $BiLSTM-SQ$ and $DPHA-C$ without regularization, against our proposed *DPHA*. The details are as follows.

- *LiteralMatching* is to match the product category vocabulary with the query words, and returns all the phrases that are both in the current query and belong to a document category. This approach is simple and widely used in many real applications. It does not consider the context information of the query or the session.
- *ClickGraph* is to generate a query representation from its clicked documents using the approach similar to [21]. We modified the approach to fit our problem that the document representations here are not bag of words, instead we use the product categories to generate the query representations. The click graph naturally maps the product categorization information from the products to the queries. In this case, as long as the user behaviors are enough, it could identify the user's intent on the query.
- $ClickGraph-VG$ is another approach proposed in [21], which is able to deal with the queries without click. It generates units (unigram, bigram, trigrams) from the clicked queries, and through click graph to generate the units' vector representations, which are product categories in our application. Then for the unseen/non-clicked queries, it estimates a regression function, to generate the queries' representations from the associated units. $ClickGraph-VG$ is able to capture the user behavior information, as well as the context information of the query.
- $BiLSTM-Q$ firstly uses a bidirectional LSTM model built on the current query's words, then predicts the multiclass classification problem of the product categories with the word embeddings. For this approach, the context information within the query is learnt by the bidirectional LSTM layer.
- $BiLSTM-SQ$ builds a bidirectional LSTM model for the entire session and the current query, and then predict the categories. This approach considers the context information in the query, as well as in the entire session.
- $DPHA-C$ is a variation of the proposed framework, which has the same structure as in Figure 4 except for the final query category prediction layer. $DPHA-C$ only uses the present query's category as output for training the model, while our proposed *DPHA* considers both the present query's category, as well as looking at the subsequent product categorization as a regularization.

6 EXPERIMENTAL RESULTS

6.1 Overall Performance

The proposed model *DPHA* is compared with several state-of-the-art baselines and the overall results are shown in Table 2. The top 3 precision, recall and F-score are reported for all the baseline approaches and the proposed framework. The *LiteralMatching* approach can find the right intent of the query only when the query is very well formed by including the categorical words literally. It is the simplest one and fastest, yet the lowest performance. The *ClickGraph* approach has showed good performance in query understanding in [21] and it outperforms *LiteralMatching* by incorporating the user behavior information towards the query. But this approach can not deal with the queries that do not have any clicks previously. The $ClickGraph-VG$ could make use of the query words information directly and generate the categories for new queries. This is a very strong baseline in query understanding from web search, and adapted to the area of product search in our application. We can see that the proposed *DPHA* framework could outperform $ClickGraph-VG$.

For the deep learning based approaches, we discuss how different type of information and the model structure could contribute to the problem of query understanding. $BiLSTM-Q$ includes the query level context information, and it outperforms *LiteralMatching* but not as well as the other approaches, and even have lower performance compared to the other query level approaches *ClickGraph*

Model	Prec@1	Recall@2	F-Score@1	Prec@@2	Recall@2	F-Score@2	Prec@@3	Recall@3	F-Score@3
LiteralMatching	0.693	0.345	0.434	0.428	0.415	0.394	0.298	0.428	0.329
ClickGraph	0.791	0.404	0.503	0.659	0.611	0.595	0.559	0.739	0.600
ClickGraph − VG	0.799	0.407	0.507	0.664	0.617	0.600	0.565	0.746	0.605
BiLSTM − Q	0.788	0.402	0.501	0.625	0.576	0.562	0.521	0.686	0.557
BiLSTM − SQ	0.808	0.413	0.514	0.638	0.589	0.574	0.527	0.694	0.564
DPHA − C	0.846	0.435	0.540	0.680	0.631	0.615	0.562	0.742	0.602
DPHA	0.849	0.436	0.542	0.684	0.635	0.618	0.566	0.746	0.606

Table 2: Comparisons of the model performance

and *ClickGraph − VG*. *BiLSTM − SQ* integrates the sessional information into the classification problem in addition to the query information, and promotes the performance compared to *BiLSTM − Q*. It indicates that the sessional information is directly quite useful to query understanding problem. On the other hand, when we look at the more refined deep framework of *DPHA − C* and *DPHA*, our designed approach show much superior results compared to using a *BiLSTM*. The hierarchical attention structure with the advantage of self-attention greatly enhance the model's performance. Compared to *DPHA−C*, using the subsequent product category in the training phrase could further benefit the effectiveness of the model.

6.2 Dynamic User Intents

Based on the manually annotated testing data set, we also analyze quantitatively how much difference the sessional information could bring to a query. For our testing set, we extract multiple sessions per query with a maximum number of 10. The average number of sessions for all queries is 8.1, since some tail queries are rare to be observed plenty of times. From the annotated testing set, we observe that the same query in different sessions could mean different intentions. In average, a query has 3.7 different meanings out of the 8.1 (47%) sessions, which means it is necessary to distinguish the query intents with the consideration of the sessional information. Figure 5 shows more details about the percentage of the sessions with different intents for the same query. The results are reported across the query frequencies (Bucket ID in the figure). A lower bucket ID means the queries have higher frequency. We can see that the top queries (in Bucket 1) relatively have more determinate meaning than the others. And the queries in Bucket 2-8 tend to be more vague, and thus more difficult to predict their categories.

Model	Precision@3		Recall@3		F-Score@3	
	NoClick	Click	NoClick	Click	NoClick	Click
LiteralMatching	0.122	0.300	0.240	0.431	0.152	0.331
ClickGraph	0.417	0.557	0.785	0.736	0.515	0.597
ClickGraph − VG	0.417	0.563	0.785	0.744	0.515	0.603
BiLSTM − Q	0.338	0.519	0.610	0.683	0.412	0.555
BiLSTM − SQ	0.366	0.525	0.690	0.691	0.452	0.562
DPHA-C	0.408	0.560	0.775	0.739	0.508	0.601
DPHA	0.437	0.564	0.803	0.743	0.537	0.604

Table 3: Performance for non-clicked and clicked <session,query>

6.3 Non-Exclusiveness of Queries' Intents

Further, we discuss the number of correct categories of queries with session-context. For all the entries <session, query> in our testing set, the average number of the correct categories is 2.4, and 74% of the entries have more than one query intent. Here only the categories generated by the proposed approaches and baselines are evaluated manually. In fact, the query intent could be even more than these reported. With the Non-Exclusiveness characteristic of the queries' intents, the query categorization problem is more challenging that the goal is to identify all possible correct categories as well as maintaining high precision.

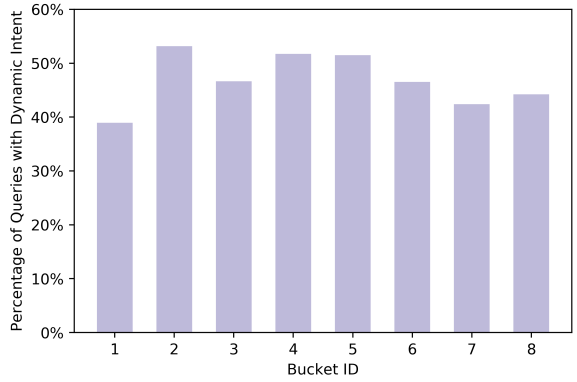


Figure 5: Percentage of Queries with Dynamic Intent in Different Sessions

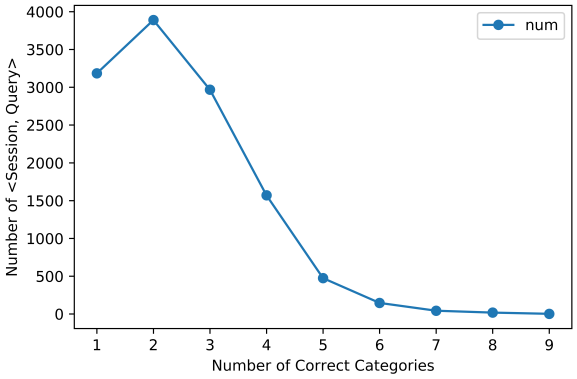
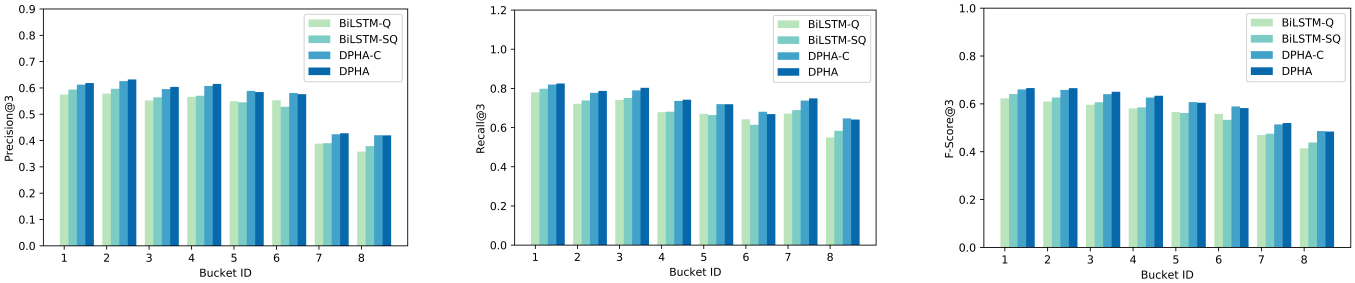


Figure 6: Number of Correct Categories for a <Session, Query> in the Testing Set



(a) Precision@3 (b) Recall@3 (c) F-Score@3

Figure 7: The performance across different frequencies of queries

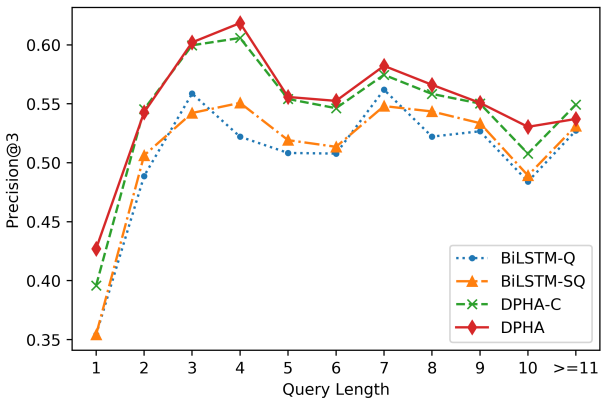


Figure 8: Precision@3 for different Query Lengths

The detailed analysis of the number of correct categories are shown in Figure 6. We can see that the most number of the testing samples have two correct categories. The maximum number of correct categories could be as high as 9. Since the majority of <session,query> are with less than or equal to 3 correct categories, we mainly focus on the evaluating the model performance at the top 3 positions.

6.4 Performance over Query Frequency

The performance on different traffics of queries is shown in Figure 7. Due to the analysis in Section 6.3, we only report the results on the top 3 positions. We can see that the performance of all approaches decreases when the bucket ID is larger, where the queries are less frequent, especially for the tail queries in bucket 7 and bucket 8. It means that the tail queries are more difficult, which aligns with our previous analysis in Section 3.2.1. This phenomenon could due to several potential reasons: The tail queries may not be well formed and thus not very informative; The tail query word context may include more noise including typos; The session context information for tail queries is not as much as the other queries. We can see from the Figure 7 that $DPHA-C$ and $DPHA$ have advantages on all the different types of queries compared to $BiLSTM-Q$ and $BiLSTM-SQ$, especiall for the less frequent queries.

6.5 Query Intent Understanding and Click Behaviors

Here we discuss the model performance in terms of click behaviors, and the results are shown in Table 3. Our testing set is splitted into two sets of <session, query> entries with and without user clicks on the present query, and the model performance are reported on the top 3 results. The sessions with clicks have higher precision compared to the non-clicked sessions. The <session, query> that results to a click behavior are more likely have the characteristics including clear query intent and well-formed context which lead to better retrieved results. On the other hand, the entries with clicks have a lower recall compared the non-clicked entries. This is due to that the non-clicked <session,query> entries have a smaller number (10% less) of correct categories compared to the clicked entries. We can see that the proposed *DPHA* is much more effective compared to the other approaches on the non-clicked queries which are more difficult. This is because *DPHA* framework has a more advanced model structure which could better understand the context information in the current query, preceding session and the subsequent category. Moreover, by better understanding the queries, it would be quite beneficial to the overall retrieval performance, and eventually boost the click through rates of the whole site.

6.6 Performance over Query Length

The query length distribution has been analysed previously in Section 3.2. We also discuss how the performance of various models change over queries with different length. The results are shown in Figure 8. In general, the *DPHA* has the best performance on all types of queries. When we take a look at the query with different lengths in detail, the precision is always lower when the queries are very short or long, and reaches the highest when the query length is around 3 or 4. For a query as short as length 1, it does not have much word-level context information. In this case, $DPHA-C$ and *DPHA* outperforms *BiLISTM* models significantly, and *DPHA* with additional regularization has more improvement over $DPHA-C$.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a new deep neural network architecture *DPHA* to model the dynamic intents of queries with respect to different scenarios. It firstly learns the representations for the

preceding queries and interested product categories with a bidirectional GRU layer with attention. Then the session's annotation is extracted from self-attention and GRU with dropouts to capture the within-session query dependencies. Both the current query and the session contributes to the prediction layer with an additional regularization from the subsequent category. We analyzed the characteristics of queries in the E-commerce domain, and conduct experiments from real-word data sets. The experimental results for query categorization show that the proposed *DPHA* model achieves better performance compared to several state-of-the-art baselines. *DPHA* is especially effective for the non-clicked queries. There are several directions for our future work. One direction is to explore other underlying factors that could enhance the query intent understanding problem. Another possible direction is to apply the proposed framework in the other vertical search domains.

REFERENCES

[1] Azin Ashkan, Charles LA Clarke, Eugene Agichtein, and Qi Guo. 2009. Classifying and characterizing query intent. In *European Conference on Information Retrieval*. Springer, 578–586.

[2] Brian Babcock, Surajit Chaudhuri, and Gautam Das. 2003. Dynamic sample selection for approximate query processing. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*. ACM, 539–550.

[3] Ricardo Baeza-Yates, Liliana Calderón-Benavides, and Cristina González-Caro. 2006. The intention behind web queries. In *International Symposium on String Processing and Information Retrieval*. Springer, 98–109.

[4] Steven M Beitzel. 2006. *On understanding and classifying web queries*. Citeseer.

[5] Steven M Beitzel, Eric C Jensen, Ophir Frieder, David Grossman, David D Lewis, Abdur Chowdhury, and Aleksandr Kolcz. 2005. Automatic web query classification using labeled and unlabeled training data. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 581–582.

[6] Andrei Z Broder, Marcus Fontoura, Evgeniy Gabrilovich, Amruta Joshi, Vanja Josifovski, and Tong Zhang. 2007. Robust classification of rare queries using web knowledge. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 231–238.

[7] Huanhuan Cao, Derek Hao Hu, Dou Shen, Daxin Jiang, Jian-Tao Sun, Enhong Chen, and Qiang Yang. 2009. Context-aware query classification. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 3–10.

[8] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. Context-aware query suggestion by mining click-through and session data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 875–883.

[9] 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).

[10] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[11] Eustache Diemert and Gilles Vandelle. 2009. Unsupervised query categorization using automatically-built concept graphs. In *Proceedings of the 18th international conference on World wide web*. ACM, 461–470.

[12] Songwei Ge, Zhicheng Dou, Zhengbao Jiang, Jian-Yun Nie, and Ji-Rong Wen. 2018. Personalizing Search Results Using Hierarchical RNN with Query-aware Attention. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 347–356.

[13] Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. 2009. Named entity recognition in query. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 267–274.

[14] Mohammed Hasanuzzaman, Sriparna Saha, Gaël Dias, and Stéphane Ferrari. 2015. Understanding temporal query intent. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 823–826.

[15] Homa B Hashemi, Amir Asiaee, and Reiner Kraft. 2016. Query intent detection using convolutional neural networks. In *International Conference on Web Search and Data Mining, Workshop on Query Understanding*.

[16] Ahmed Hassan, Xiaolin Shi, Nick Craswell, and Bill Ramsey. 2013. Beyond clicks: query reformulation as a predictor of search satisfaction. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. ACM, 2019–2028.

[17] Yunlong He, Jiliang Tang, Hua Ouyang, Changsung Kang, Dawei Yin, and Yi Chang. 2016. Learning to rewrite queries. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1443–1452.

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

[19] 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.

[20] Jyun-Yu Jiang, Yen-Yu Ke, Pao-Yu Chien, and Pu-Jen Cheng. 2014. Learning user reformulation behavior for query auto-completion. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 445–454.

[21] 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 *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 185–194.

[22] Madian Khabsa, Zhaohui Wu, and C Lee Giles. 2016. Towards better understanding of academic search. In *2016 IEEE/ACM Joint Conference on Digital Libraries (JCDL)*. IEEE, 111–114.

[23] Mu-Chu Lee, Bin Gao, and Ruofei Zhang. 2018. Rare query expansion through generative adversarial networks in search advertising. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 500–508.

[24] Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. (2015).

[25] Bhaskar Mitra. 2015. Exploring session context using distributed representations of queries and reformulations. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. ACM, 3–12.

[26] Mahesh Chandra Mukkamala and Matthias Hein. 2017. Variants of rmsprop and adagrad with logarithmic regret bounds. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2545–2553.

[27] José Luis Ortega and Isidro Aguillo. 2010. Differences between web sessions according to the origin of their visits. *Journal of Informetrics* 4, 3 (2010), 331–337.

[28] Rishiraj Saha Roy, Rahul Katare, Niloy Ganguly, Srivatsan Laxman, and Monojit Choudhury. 2015. Discovering and understanding word level user intent in web search queries. *Journal of Web Semantics* 30 (2015), 22–38.

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

[30] Yangyang Shi, Kaisheng Yao, Le Tian, and Daxin Jiang. 2016. Deep lstm based feature mapping for query classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1501–1511.

[31] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 553–562.

[32] K Sreelakshmi, PC Rafeeque, S Sreetha, and ES Gayathri. 2018. Deep Bi-Directional LSTM Network for Query Intent Detection. *Procedia computer science* 143 (2018), 939–946.

[33] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[34] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[36] Hongli Wang and Jiangtao Ren. 2018. A Self-Attentive Hierarchical Model for Jointly Improving Text Summarization and Sentiment Classification. In *Asian Conference on Machine Learning*. 630–645.

[37] Zhongyuan Wang, Kejun Zhao, Haixun Wang, Xiaofeng Meng, and Ji-Rong Wen. 2015. Query understanding through knowledge-based conceptualization. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[38] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

[39] Minoru Yoshida, Shin Matsushima, Shingo Ono, Issei Sato, and Hiroshi Nakagawa. 2010. ITC-UT: Tweet Categorization by Query Categorization for On-line Reputation Management. In *CLEF*, Vol. 170. Citeseer.