Online Bayesian Learning for E-Commerce Query Reformulation

Gaurush Hiranandani
University of Illinois at Urbana-Champaign
gaurush2@illinois.edu

Sumeet Katariya
Amazon Inc.
katsumee@amazon.com

Nikhil Rao
Amazon Inc.
nikhilsr@amazon.com

Kathik Subbian
Amazon Inc.
ksubbian@amazon.com

1 Introduction

Customers search for products on an e-commerce website by entering a query, and the search engine returns products which best match the query. In addition to the product metadata such as titles and description, the search engine relies heavily on past behavioral signals (clicks, purchases, etc.) to retrieve the best set of items. The performance of the search engine is usually good on *head* (high frequency) queries due to the rich availability of historical behavioral signals on these queries. On the tail (low frequency) queries, however, the performance can be much worse compared to the head due to the search engine's over-reliance on the behavioral data. Given that real world user query distributions are fat-tailed [2], this affects a significant fraction of queries, underlining the need to design methods that allow the search engine to learn well on the tail. One way to address this issue is to reformulate a tail query into an appropriate head query that the search engine is attuned to and which also preserves the *purchase intent* of the tail query.

The key challenge in mapping a tail query to a head query is preserving the customer's purchase intent. From the search engine's perspective, two queries should be considered equivalent if they lead to the purchase of the same or similar set of products. This property can be used to define a similarity metric over the space of queries, which can then be used to learn a representation for queries using a deep neural network (DNN). However, this similarity metric is noisy for tail queries because a) tail queries are rare, and b) there is little to no information about products purchased in response to tail queries, because the search engine may retrieve many irrelevant products for these queries. This chicken-and-egg problem can result in sub-optimal representations for tail queries [6].

We address this issue by using Bayesian contextual bandit techniques which refine the representations of the tail queries from the DNN without severely affecting the user experience. In particular, we explore the space of head queries that a tail query can be mapped to, and use the customer's actions in response to the reformulated head query as reward to fine-tune the DNN in an online low-regret fashion. We make use of Thompson Sampling as well as a nonlinear variant (Bayes by Backprop [1]) to map head and tail query embeddings into a common Euclidean space.

Our contributions are as follows: We define a purchase similarity metric over queries and use this to learn query representations (embedding) that aligns with their purchase intent. Second, we propose and formulate a contextual multi-armed bandit problem to explore representations for tail queries where representation learning is the hardest. Third, we propose two practical Bayesian online learning algorithms for the query reformulation task and evaluate them on synthetic and real-world datasets.

Notation: While the methods we propose are motivated by mapping tail queries to head, these are highly generic and can be used to reformulate queries from any distribution, to a query that lies in a large but predefined set (head queries in our case). To this end, we will interchangeably refer to the tail query as “source" and reformulated head query as “target" in the rest of the paper.

4th workshop on Bayesian Deep Learning (NeurIPS 2019), Vancouver, Canada.

2 Problem Formulation and Implementation

Let $\mathcal{H}$ and $\mathcal{S}$ represent the sets of head and tail queries respectively, and let $\gamma : (\mathcal{H} \cup \mathcal{S}) \times (\mathcal{H} \cup \mathcal{S}) \to [0, 1]$ be a function that measures the similarity of two queries. We assume that $\gamma(h_1, h_2)$ is known for any $h_1, h_2 \in \mathcal{H}$. The interaction is modeled as follows: at time t, the environment (customer) chooses a query $s_t \in \mathcal{S}$, and the learning agent returns a query $h_t \in \mathcal{H}$. The learning agent receives a reward $r_t \sim \mathcal{D}(\gamma(s_t, h_t))$, where $\mathcal{D}$ is any suitable distribution with mean $\gamma(s_t, h_t)$. We define $h_t^* = \arg\max_{h \in \mathcal{H}} \gamma(s_t, h)$ to be the query with the highest similarity to s_t, and measure the performance of the agent in terms of its expected cumulative regret in n steps as

$$R(n) = \sum_{t=1}^{n} \mathbb{E}[\gamma(s_t, h_t^*) - r_t] = \sum_{t=1}^{n} \mathbb{E}[\gamma(s_t, h_t^*) - \gamma(s_t, h_t)], \tag{1}$$

where the expectation is taken over the randomness in the choice of s_t and the reward r_t.

Implementation Details: (a) For a query $h \in \mathcal{H}$, the relative purchases across all the products provides us a purchase distribution $P(h)$ for h. We then model $\gamma(h_1, h_2) = \langle P(h_1), P(h_2) \rangle$, where $\langle \cdot, \cdot \rangle$ denotes a dot product, and refer it as the *purchase-similarity*. (b) In the online setting, the reward r_t for the learning agent is set as follows. When a customer enters the query $s_t \in \mathcal{S}$, the search engine retrieves items corresponding to a reformulated head query h_t, and we set $r_t = 1$ if the customer engages (clicks/purchases) with the retrieved products, and 0 otherwise.

3 Algorithms

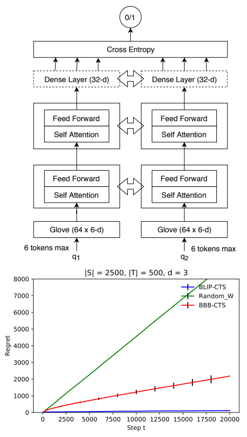
We use a siamese transformer-based [9] deep neural network (DNN) that takes as input a pair of queries (in the form of GLOVE embedding [4]) and outputs a binary label denoting whether the two queries are purchase-similar. We initially pretrain the model using pairs of head queries from $\mathcal{H}$, with ground truth labels generated as $\mathbf{1}[\langle P(h_i), P(h_j) \rangle \geq \tau]$, where τ is a pre-defined threshold, and $\mathbf{1}[\cdot]$ is the indicator function. This pretraining learns a d-dimensional, purchase-similar representation of the queries (the last dense layer in Figure 1(a)) and allows us to initialize a reasonable embedding space to map tail queries to the head. Here on, we use s_t to denote both the query as well as its d-dimensional representation. We next describe two bayesian bandit methods to refine this embedding. The derivation of the update equations for both algorithms is provided in Appendix A.

Bayesian Linear Probit Contextual Thompson Sampling (BLIP-CTS) : BLIP-CTS performs bayesian generalized linear regression [3] on top of the representation of the last layer of the transformer DNN [5, 8]. It models the similarity between a source query s_t and a head query h_t by assuming that the engagement reward r_t is sampled from a distribution with

$$\mathbb{P}(r_t = 1 | s_t, h_t; W_*) = \phi\left(\frac{\langle h_t, W_* s_t \rangle}{\beta}\right), \tag{2}$$

where W_* is an unknown matrix, and ϕ denotes the standard Gaussian CDF. Intuitively, (2) warps the source query s_t onto the space of head queries using the matrix W_*, thus learning the proper aligment between head and tail query spaces. We assume that the entries in W_* are drawn independently from a Gaussian distribution with parameters μ_{ij}, σ_{ij}. BLIP-CTS maintains a posterior distribution $\mathcal{W}_t$ over W_*. At time t, it samples $W_t \sim \mathcal{W}_t$ and selects the head query $h_t = \arg\max_{h \in \mathcal{H}} \langle h, W_t s_t \rangle$ as the reformulation of s_t. The mean and variance parameters $\{\mu_{ij}, \sigma_{ij}\}_{i,j=1}^{d}$ of the distribution $\mathcal{W}_t$ are updated using the observed reward r_t (lines 7-9 in Algorithm 1).

Bayes By Backprop Contextual Thompson Sampling (BBB-CTS) : Bayes by Backprop [1] finds a distribution from a tractable family that minimizes the KL divergence to the posterior distribution over the weights of a neural network. Unlike BLIP-CTS where a neural network with fixed weights is trained over head queries and exploration is done through bayesian linear regression on the last layer only, BBB maintains a distribution over all the weights of the neural network. BBB-CTS samples a DNN from the approximate posterior, and for a given source query s_t selects the head query h_t that maximizes similarity as measured by the sampled neural network. We can see that BBB-CTS is a nonlinear variant of the BLIP-CTS method described above.



```
Algorithm 1: BLIP-CTS
1. Input: Parameters $\beta > 0$, $\mu_0, \sigma_0^2 \in \mathbb{R}^{d\times d}$
2. For  $t = 1, 2, \cdots, T$ do
3.     Sample $W_t \sim \mathcal{N}(\mu_{t-1}, \sigma_{t-1}^2)$
4.     Observe context $s_t$ (a source query).
5.     Choose optimal action $h_t = \arg\max_{h\in\mathcal{H}} h^T W_t s_t$.
6.     Observe reward by sampling $r_t \sim \phi\left(\frac{h_t^T W_* s_t}{\beta}\right)$
7.     Set $\delta^2 = \beta^2 + (h_t \odot h_t)^T \sigma_{t-1}^2 (s_t \odot s_t)$
8.     Set $\mu_t = \mu_{t-1} + \frac{r_t}{\delta}\nu\left(\frac{r_t h_t^T \mu_{t-1} s_t}{\delta}\right)\left[h_t s_t^T \odot \sigma_{t-1}^2\right]$
9.     Set $\sigma_t^2 = \sigma_{t-1}^2\left[1 - \frac{1}{\delta^2}\omega\left(\frac{r_t h_t^T \mu_{t-1} s_t}{\delta}\right)\left[(h_t s_t^T \odot h_t s_t^T) \odot \sigma_{t-1}^2\right]\right]$,
       where $\nu(z) = \frac{\mathcal{N}(z;0,1)}{\phi(z;0,1)}$ and $\omega(z) = \nu(z)(\nu(z) + z)$.
10. Output: $\hat{\mu}$, $\hat{\sigma}^2$. For inference, we take the final matrix $\hat{W} = \hat{\mu}$
```

Figure 1: (a) The siamese transformer model trained for query embedding (left top). (b) BLIP-CTS and BBB-CTS comparison on simulations (left bottom). (c) Algorithm 1 for BLIP-CTS (right).

4 Experiments

Simulations: We first compare the algorithms in a simulated setting, where the reward model follows (2). We take two sets $\mathcal{S}$ (sources), $|\mathcal{S}| = 2500$ and $\mathcal{H}$ (targets), $|\mathcal{H}| = 500$, where each $x \in \mathcal{S} \cup \mathcal{H}$ is a 3-dimensional vector generated from standard Gaussian distribution. We define a $W_* \in \mathcal{R}^{3\times 3}$, which is unknown to the learning agent. We then compare BLIP-CTS and BBB-CTS in terms of the regret (1). The regret averaged over 10 runs is reported in Figure 1(b). We observe that BLIP-CTS outperforms BBB-CTS. This is expected, since we assume a linear reward model as in (2). Experiments with non-linear reward models are left for the future. Since BLIP-CTS significantly outperforms BBB-CTS, for now, we use BLIP-CTS to reformulate queries from a real dataset.

Real-World Experiments: We use a dataset of anonymized user logs from one month of a popular e-commerce website. We create a set of head queries $\mathcal{H}$ by considering queries that have been searched more than 10 times, yielding $|\mathcal{H}| = 6.2M$ head queries. For query reformulation, we are specifically interested in those queries that are very rare and have few or no behavioral information (clicks, purchases, etc.) from users. Since the number of such queries is huge, we take only 0.5% of queries that have been searched at most twice, giving us $1.3M$ tail (source) queries. We pretrain the siamese transformer network on $\mathcal{H}$, with $\tau = 0.01$, sampling random negatives, and using the cross-entropy loss. All hyperparameters are tuned on a held out validation set. The transformers have 2 attention layers with mean-pooling followed by a dense layer to yield 32-dimensional query embeddings. We call this model M_0.

Next, we refine the embeddings from M_0 via Algorithm 1, and call this new model M_1. For prototyping purposes we use a "proxy" oracle to yield rewards. Specifically, we use a classifier $f(\cdot)$ trained on a separate set of human annotated queries with their product category provided. $f(\cdot)$ takes as input a query, and returns it's product category. For our reward mechanism, we sample $r_t \sim \phi((f(s_t) == f(h_t))/\beta)$. We choose this oracle as $f(\cdot)$ has been found to achieve a high product category classification accuracy, and identifying product category provides a considerable information regarding the purchase intent for an e-commerce query [7]. However, since this oracle has persistent noise in the reward mechanism (i.e. $f(\cdot)$ can make deterministic errors), we perform majority voting via multiple source and head queries before updating the statistics in BLIP-CTS. Moreover, since this is a pseudo-oracle, the best action for a source query s is chosen via: $h_t = \arg\max_{h\in\mathcal{H}} h \cdot (I + \lambda\hat{W})s_t$, where λ is cross-validated and controls the alignment of embedding space through product category matches, and $\hat{W}$ denotes the matrix learned by BLIP-CTS.

We report some results for $\lambda = 0.2$ in Table 1. Please see Table 2 in Appendix B for more results. We see that BLIP-CTS captures the purchase intent behind the source queries remarkably well in comparison to M_0. For Source 1, though M_0 captures most of the intent behind the query, the refinement in BLIP-CTS helps achieve much better results. In Source 2, M_0 suggests outdoor trash **can** related items (the transformer embedding is inaccurate); whereas, BLIP-CTS suggests trash **bags** while retaining the context that it is needed for outdoor purposes (via careful exploration). In Source 3, M_0 misunderstands the user intent, while BLIP-CTS correctly narrows down to auto sun shades.

Table 1: Qualitative evaluation of the methods. We show the nearest five neighbors for a given source query. Our proposed (BLIP-CTS) method outperforms the baseline (M_0) on multiple source queries.

M_0	M_1 (BLIP-CTS)	M_0	M_1 (BLIP-CTS)
Source 1: under armor underw		Source 2: keter outdoor trash can trash bags	
under armour wwp	under armour tech boxerjock	hefty step garbage can	outdoor trash bags
under armour barren	under armour magnetico pro	outdoor trash can storage shed	black outdoor trash bags
under armour sportstyle	under armour tech	outside trash bin	hefty step garbage can
under armour breathelux	under armour ua tech 2.0	outside trash bin storage	large outdoor trash bags
under armour culver	under armour tech 2.0	outside trash can storage	grow bags 30 gallon with handles
Source 3: auto shade, land rover discovery		Source 4: cat6a 32ft	
las vegas sail boat	dodgers sunshade for cars	cat6a	cat6a 200ft
shade shore	ohio state car mats	cat6a 200ft	cat6a plenum
shade & shore	car cover outside land rover lr3	cat6a plenum	cat6a
a shade of vampire 77	carolina skiff boat cover	vandesail cat7	cat6a 500ft
carolina skiff boat cover	detroit lions car mats	ftdi ttl-232r-3v3	cat6a 1000ft

For Source 4, we see that BLIP-CTS results in more relevant query reformulations even down the order when compared to M_0.

5 Conclusions and Future Work

We conclude that the proposed purchase-similarity metric over queries helps to learn a reasonable query embedding that aligns with purchase intent. We propose two Bayesian online learning algorithms, BLIP-CTS and BBB-CTS, to refine the embeddings for tail queries. We observe that BLIP-CTS performs better than the initial model trained on signals from head queries alone. We are working to deploy this model and replace the pseudo-reward by an engagement based reward. This reward contains much more information than just the product category match. Note that in our simulations, BLIP-CTS is expected to outperform BBB-CTS since the reward is modeled by (2). We conjecture that under complex reward models, BBB-CTS will show better query reformulation performance in comparison to BLIP-CTS. Therefore, we plan to study alternate non-linear reward models and evaluate BBB-CTS under these settings. Proving theoretical guarantees for BBB-CTS is also an interesting research direction for the future.

References

[1] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *ICML*, pages 1613–1622, 2015.

[2] Sharad Goel, Andrei Broder, Evgeniy Gabrilovich, and Bo Pang. Anatomy of the long tail: ordinary people with extraordinary tastes. In *WSDM*, pages 201–210. ACM, 2010.

[3] Thore Graepel, Joaquin Quiñonero Candela, Thomas Borchert, and Ralf Herbrich. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *ICML*, pages 13–20, 2010.

[4] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543, 2014.

[5] Carlos Riquelme, George Tucker, and Jasper Snoek. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. *arXiv preprint arXiv:1802.09127*, 2018.

[6] George Karypis Saurav Manchanda, Mohit Sharma. Intent term selection and refinement in e-commerce queries. In *CIKM*, pages 5998–6008, 2017.

[7] Gyanit Singh, Nish Parikh, and Neel Sundaresan. Rewriting null e-commerce queries to recommend products. In *WWW*, pages 73–82. ACM, 2012.

[8] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *ICML*, pages 2171–2180, 2015.

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

Appendices

A Derivation of Updates

A.1 BLIP-CTS Algorithm Updates

We assume that the probability model governing the click/purchase (label) is a probit link function. Let us denote the label (click or no-click) by r. Therefore,

$$\mathbb{P}(r|s_t, h_t; W) = \phi\left(\frac{r\langle h_t, Ws_t\rangle}{\beta}\right), \tag{3}$$

where $\phi(z) = \int_{-\infty}^{z} \mathcal{N}(a)da$ is the cumulative distribution function for a standard normal random variable. The parameter β scales the steepness of the distribution. We also assume that the prior distribution over W is the product of d independent normal random variables,

$$\mathbb{P}(W) = \Pi_{i=1}^{d}\Pi_{j=1}^{d}\mathcal{N}(W_{ij}; \mu_{ij}, \sigma_{ij}) \tag{4}$$

parametrized by a mean matrix $\mu \in \mathbb{R}^{d\times d}$ and a variance matrix $\sigma \in \mathbb{R}^{d\times d}$.

We sequentially iterate over the data and update the distribution of the weight matrix observation for a given data set $\{(h_t, s_t, r_t)\}_{t=1}^{T}$. For ease of notation we denote the outcome probability for data point at time t by

$$v_t(W) := \mathbb{P}(r_t|W, s_t, h_t). \tag{5}$$

In each iteration, we form the true posterior distribution and then approximate it with the best matching chosen parametric distribution. In BLIP-CTS, the approximating distribution is chosen to be a product of joint independent normal random variables and is found through minimizing the Kullback-Liebler (KL) divergence. In particular, the following two steps are performed in each iteration:

1. Denote the prior distribution over W with mean and variance matrix (μ_t, σ_t^2) by

$$q_t(W) := \Pi_{i=1}^{d}\Pi_{j=1}^{d}\mathcal{N}(W_{t_{ij}}; u_{t_{ij}}, \sigma_{t_{ij}}^2) \tag{6}$$

   and the posterior distribution given (h_t, s_t, r_t) by $\hat{p}_t$ using the Bayes rule:

$$\hat{p}_t(W|h_t, s_t, r_t) := \frac{v_t(W)q_t(W)}{\int_W v_t(W)q_t(W)dW}. \tag{7}$$

2. Find the closest independent normal approximation

$$\hat{q}_t(W) = \Pi_{i=1}\Pi_{j=1}\mathcal{N}(W_{ij}; \hat{\mu}_{ij}, \hat{\sigma}_{ij}^2) \tag{8}$$

   to the posterior (7) by minizming the KL divergence:

$$\mu_{t+1}, \sigma_{t+1}^2 = \underset{\hat{\mu}, \hat{\sigma}^2}{\arg\min}\ KL(\hat{p}_t(W|h_t, s_t, r_t)||\hat{q}(W)). \tag{9}$$

Next, we discuss the solution of the above minimization problem. For ease of notation, we will remove the subscript t and instead use the subscript “new" to denote the parameters for the posterior distribution $(\mu_{new}, \sigma_{new}^2)$. The KL divergence from the approximate normal distribution $\hat{q}$ to the exact posterior distribution $\hat{p}$ is

$$\begin{aligned} F(\mu_{new}, \sigma_{new}^2) &:= KL(\hat{p}(W|h, s, y)||\hat{q}(W|\mu_{new}, \sigma_{new}^2)) \\ &= \int_W (\hat{p}\log\hat{p} - \hat{p}\log\hat{q})dW. \end{aligned}$$

While minimizing F, we only need to focus the second part in the integral, because the first part is not a function of the new parameters. From (8), we have

$$\log\hat{q} = -\frac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{d}\log 2\pi\sigma_{new_{ij}}^2 + \sum_{i=1}^{d}\sum_{j=1}^{d}\frac{(W_{ij} - \mu_{new_{ij}})^2}{\sigma_{new_{ij}}^2}. \tag{10}$$

Thus, the parameters for the posterior distribution are

$$\mu_{new}, \sigma^2_{new} = \arg\min_{\mu,\sigma^2} \left\{ \frac{1}{2} \int_W \hat{p} \left(\sum_i \sum_j \log 2\pi\sigma^2_{ij} + \sum_{i=1} \sum_{j=1} \frac{(W_{ij} - \mu_{ij})^2}{\sigma^2_{ij}} dW \right) \right\}. \quad (11)$$

Setting $\frac{\partial F}{\partial \mu_{ij}} = 0$ yields,

$$\mu_{new_{ij}} = \int_W \hat{p} W_{ij} dW = \mathbb{E}_{\hat{p}}[W_{ij}]. \quad (12)$$

Setting $\frac{\partial F}{\partial \sigma^2_{ij}} = 0$ yields,

$$\sigma^2_{new_{ij}} = \int_W \hat{p}(W_{ij} - \mu_{new_{ij}})^2 dW = \mathbb{E}_{\hat{p}}[(W_{ij} - \mu_{new_{ij}})^2]. \quad (13)$$

The Hessian of F at the above mean and variance estimates comes out to be positive definite as well (the details will be shown in an extended version of the paper); therefore, satisfying the sufficient condition for them being the minimizer for F.

Now, let the normalizing constant in (7) be $Z \coloneqq \int_W v(W)q(W)dW$.

$$\begin{aligned}
\frac{\partial \log Z}{\partial \mu_{ij}} &= \frac{1}{Z} \frac{\partial Z}{\partial \mu_{ij}} \\
&= \frac{1}{Z} \int_W v(W) \frac{\partial q}{\partial \mu_{ij}}(W) dW \\
&= \frac{1}{Z} \int_W v(W) \left(\Pi^d_{m=1, m\neq i} \Pi^d_{n=1, n\neq j} \mathcal{N}(W_{mn}; \mu_{mn}, \sigma^2_{mn}) \right) \frac{\partial \mathcal{N}(W_{ij}; \mu_{ij}, \sigma^2_{ij})}{\partial \mu_{ij}} dW \\
&= \int_W \frac{v(W)q(W)}{Z} \left(\frac{W_{ij} - \mu_{ij}}{\sigma^2_{ij}} \right) dW \\
&= \frac{1}{\sigma^2_{ij}} (\mathbb{E}_{\hat{p}}[W_{ij} - \mu_{ij}])
\end{aligned}$$

Putting this in (12), we get

$$\mu_{new_{ij}} \coloneqq \mu_{ij} + \sigma^2_{ij} \frac{\partial \log Z}{\partial \mu_{ij}}. \quad (14)$$

Similarly, taking the derivative of $\log Z$ w.r.t the σ^2_{ij} yields,

$$\begin{aligned}
\frac{\partial \log Z}{\partial \sigma^2_{ij}} &= \frac{1}{Z} \int_W v(W) \left(\Pi^d_{m=1, m\neq i} \Pi^d_{n=1, n\neq j} \mathcal{N}(W_{mn}; \mu_{mn}, \sigma^2_{mn}) \right) \frac{\partial \mathcal{N}(W_{ij}; \mu_{ij}, \sigma^2_{ij})}{\partial \sigma^2_{ij}} dW \\
&= \frac{1}{Z} \int_W v(W) \left(\Pi^d_{m=1, m\neq i} \Pi^d_{n=1, n\neq j} \mathcal{N}(W_{mn}; \mu_{mn}, \sigma^2_{mn}) \right) \mathcal{N}(W_{ij}; \mu_{ij}, \sigma^2_{ij}) \times \\
&\qquad \times \left(-\frac{1}{2\sigma^2_{ij}} + \frac{1}{2} \frac{(W_{ij} - \mu_{ij})^2}{\sigma^4_{ij}} \right) dW \\
&= \int_W \hat{p}(W) \left(-\frac{1}{2\sigma^2_{ij}} + \frac{1}{2} \frac{(W_{ij} - \mu_{ij})^2}{\sigma^4_{ij}} \right) \\
&= -\frac{1}{2\sigma^2_{ij}} + \frac{1}{2\sigma^4_{ij}} (\mathbb{E}_{\hat{p}}[W^2_{ij}] - 2\mu_{new_{ij}} \mu_{ij} + \mu^2_{ij}) \\
&= -\frac{1}{2\sigma_{ij}} + \frac{1}{2\sigma^4_{ij}} \left(\mathbb{E}_{\hat{p}}[W^2_{ij}] - (\mu_{new_{ij}})^2 + \sigma^4 \left(\frac{\partial \log Z}{\partial \mu_{ij}} \right)^2 \right)
\end{aligned}$$

Since $\mathbb{E}_{\hat{p}}[W_{ij}^2] - (\mu_{new_{ij}})^2$ is $\sigma^2_{new_{ij}}$, the update equations become:

$$\sigma^2_{new_{ij}} = \sigma^2_{ij} + \sigma^4_{ij}\left(2\frac{\partial Z}{\partial \sigma^2_{ij}} - \left(\frac{\partial \log Z}{\partial \mu_{ij}}\right)^2\right). \tag{15}$$

Update equations (14) and (15) are independent of the reward model assumptions. When the reward is taken to be probit link function as described in (3), then it is easy to re-write the above update equations as follows:

$$\begin{aligned}
\delta^2 &= \beta^2 + (h \odot h)^T \sigma^2 (s \odot s)\\
\mu_{new} &= \mu + \frac{r}{\delta}\nu\left(\frac{rh^T \mu s}{\delta}\right)\left[hs^T \odot \sigma^2\right]\\
\sigma^2_{new} &= \sigma^2\left[1 - \frac{1}{\delta^2}\omega\left(\frac{rh^T\mu s}{\delta}\right)\left[(hs^T \odot hs^T)\odot \sigma^2\right]\right],
\end{aligned}$$

where $\nu(z) = \frac{\mathcal{N}(z;0,1)}{\phi(z;0,1)}$ and $\omega(z) = \nu(z)(\nu(z)+z)$, and $\odot$ denotes the Hadamard product. These are the update equations in lines 7-9 in Algorithm 1.

A.2 BBB-CTS Algorithm with Parameter Updates

In this section, we discuss the application of Bayes by Backprop algorithm [1] in the contextual bandit based query reformulation problem. We assume a posterior distribution over weighs W of a neural network,

$$\mathbb{P}(W|\mathcal{D}) = \frac{\mathbb{P}(\mathcal{D}|W)\mathbb{P}(W)}{\mathbb{P}(\mathcal{D})} = \frac{\mathbb{P}(\mathcal{D}|W)\mathbb{P}(W)}{\int_W \mathbb{P}(\mathcal{D}|W)\mathbb{P}(W)dW}, \tag{16}$$

where $\mathcal{D}$ is the the training data occurring sequentially with time in the form of a set $\{(h,s,r)_t\}$, $\mathbb{P}(W|\mathcal{D})$ is the posterior probability of W, $\mathbb{P}(\mathcal{D}|W)$ is the likelihood of W, $\mathbb{P}(W)$ is the prior probability on W, and $\mathbb{P}(\mathcal{D})$ is the evidence of the data. Furthermore, the inference is done by taking the weighted expectation over all possible values of W:

$$\mathbb{P}(\hat{r}|h,s) = \mathbb{E}_{\mathbb{P}(W|\mathcal{D})}[\mathbb{P}(\hat{r}|h,s,W)] = \int \mathbb{P}(\hat{r}|h,s,W)\mathbb{P}(W)dW, \tag{17}$$

where $\mathbb{P}(\hat{r}|h,s,W)$ is the conditional probability of the reward (click/purchase) given h, s, W. Variational inference involves constructing a new variational posterior distribution $q(W|\theta)$ with parameters θ, that approximates the true posterior $\mathbb{P}(W|\mathcal{D})$ i.e.:

$$\theta^* = \arg\min_\theta \mathrm{KL}[q(W|\theta)\|P(W|\mathcal{D})].$$

The resulting loss function is:

$$\begin{aligned}
\mathcal{F}(\mathcal{D},\theta) &= \int q(W|\theta)\log\frac{q(W|\theta)}{P(W)} - q(W|\theta)\log P(\mathcal{D}|W)\mathrm{d}W \qquad (18)\\
&= \mathrm{KL}[q(W|\theta)\|P(W)] - \mathbb{E}_{q(W|\theta)}[\log P(\mathcal{D}|W)]. \qquad (19)
\end{aligned}$$

Computing the expectation of the likelihood over the variational posterior is computationally intractable, so we approximate the cost function using sampled weights as follows:

$$\mathcal{F}(\mathcal{D},\theta) \approx \sum_{i=1}^{n} \log q(\mathbf{w}^{(i)}|\theta) - \log P(\mathbf{w}^{(i)}) - \log P(\mathcal{D}|\mathbf{w}^{(i)}). \tag{20}$$

Thus, one can now use the automatic differentiation provided by tools such as TensorFlow or PyTorch to minimize this function. We only look into the sampling of weights and setting up the cost function

```
Algorithm 2: BBB-CTS
1. Input: Parameters $\beta > 0$, $\mu_0, \rho_0 \in \mathbb{R}^{d\times d}$, learning rate $\alpha$
2. For  $t = 0, 1, \cdots, T$ do
3.      Sample $\epsilon_t \sim \mathcal{N}(0, I)$
4.      Let $W_t = \mu_t + \log(1 + \exp(\rho_t)) \odot \epsilon_t$
5.      Let $\theta_t = (\mu_t, \rho_t)$
6.      Observe a source query $s_t \in \mathcal{S}$.
7.      Choose optimal action $h_t = \arg\max_{h\in\mathcal{H}} h^T W_t s_t$ (following the reward model in (2)).
8.      Observe reward by sampling $r_t \sim \phi\left(\frac{h_t^T W_* s_t}{\beta}\right)$
9.      Let $f(W_t, \theta_t) = \log q(W_t|\theta_t) - \log \mathbb{P}(W_t)\mathbb{P}(\mathcal{D}|W_t)$, where $\mathcal{D} = \{h_k, s_k, r_k\}_{k=0}^{t}$
10.     Calculate the gradient with respect to the mean $\mu$: $\nabla\mu = \frac{\partial f(W,\theta)}{\partial W} + \frac{\partial f(W,\theta)}{\partial \mu}$
11.     Calculate the gradient with respect to the standard deviation parameter $\rho$:
          $\nabla\rho = \frac{\partial f(W,\theta)}{\partial W} \frac{\epsilon}{1+\exp(-\rho)} + \frac{\partial f(W,\theta)}{\partial \rho}$
12.     Update the variational parameters: $\mu \leftarrow \mu - \alpha\nabla\mu$, $\rho \leftarrow \rho - \alpha\nabla\rho$.
13. Output: $\mu_T$, $\sigma_T^2$. For inference, we take the final matrix $\hat{W} = \hat{\mu}_T$
```

Table 2: Qualitative evaluation: We show the nearest five neighbors for a given source query. Our proposed BLIP-CTS model (M_1) outperforms the baseline (M_0) on multiple source queries.

M_0	M_1 (BLIP-CTS)	M_0	M_1 (BLIP-CTS)
Source 1: under armor underw		Source 2: keter outdoor trash can trash bags	
under armour wwp	under armour tech boxerjock	hefty step garbage can	outdoor trash bags
under armour barren	under armour magnetico pro	outdoor trash can storage shed	black outdoor trash bags
under armour sportstyle	under armour tech	outside trash bin	hefty step garbage can
under armour breathelux	under armour ua tech 2.0	outside trash bin storage	large outdoor trash bags
under armour culver	under armour tech 2.0	outside trash can storage	grow bags 30 gallon with handles
Source 3: auto shade, land rover discovery		Source 4: cat6a 32ft	
las vegas sail boat	dodgers sunshade for cars	cat6a	cat6a 200ft
shade shore	ohio state car mats	cat6a 200ft	cat6a plenum
shade & shore	car cover outside land rover lr3	cat6a plenum	cat6a
a shade of vampire 77	carolina skiff boat cover	vandesail cat7	cat6a 500ft
carolina skiff boat cover	detroit lions car mats	ftdi ttl-232r-3v3	cat6a 1000ft
Source 5: tye die use shirt		Source 6: rug dig bed	
tye die tshirt	tye die shirt	rug foam	under bed rug
tye die shirt	tye die tshirt	bed rug	rug for under bed
state line tack	tie die shirt kit	rug for bed	bed rug
tye die shirts	tye die shirts	rug for under bed	aqua throw rug
tie die shirt kit	state line tack	under bed rug	rug under bed
Source 7: oneplus 5 sticker skin		Source 8: 2006 prius rubber mats	
gameboy sticker	gameboy sticker	toyota corolla rubber floor mats	dodge ram rubber floor mats
lightening mcqueen stickers	airpod sticker skin	2002 f250 floor mats	toyota corolla rubber floor mats
final fantasy 7 decal	lightening mcqueen stickers	honda accord rubber floor mats	toyota tacoma rubber floor mats
paint by number anime	iphone x sticker skin	2010 f150 floor mats supercrew	honda accord rubber floor mats 500ft
stickers 400 pcs	state line tack	2009 camry floor mats	2002 f150 floor mats
Source 9: atoms sound bar		Source 10: dark green twill tape	
gogroove sound bar	megacra sound bar	dark green tape	dark green tape
nakamichi sound bar	wetsound sound bar skin	luminous tape	dark brown tape
wetsounds sound bar	kuryakyn sound bar	od green tape	dark brown duck tape
boes sound bar	meidong sound bar	blue hockey tape	dark tape
naxa sound bar	zvox sound bar	florist tape green	od green tape

as above. We can then leverage the usual backpropagation methods to train a model. It is found to be good approximation for diagonal Gaussian distribution [1], i.e.:

$$\theta = (\mu, \rho), \sigma = \log(1 + e^{\rho}), P(W) = \prod_{i,j} \mathcal{N}(W_{ij}|0, \sigma^2).$$

Following the above formulation, in Algorithm 2, we discuss the online learning algorithm BBB-CTS for query reformulation including the update rules.

B Extended Results

In Table 2, we show some additional results comparing M_0 and BLIP-CTS. We see that the embedding from BLIP-CTS gives much better results for query reformulation in comparison to M_0.