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. An over-reliance on past behavioral signals can potentially impact the performance on the tail (low frequency) queries, where there is a lack of behavioral data. Given that real world user query distributions are fat-tailed, 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) the performance of the search engine for these queries is poor, hence there is little to no information about products purchased in response to these queries. This chicken-and-egg problem can result in sub-optimal representations for tail queries [5].

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.

2 Problem Formulation and Implementation

Let $\mathcal{H}$ and $\mathcal{S}$ represent the sets of head and tail queries respectively, and let $\mu : (\mathcal{H} \cup \mathcal{S}) \times (\mathcal{H} \cup \mathcal{S}) \to [0, 1]$ be a function that measures the similarity of two queries. We assume that $\mu(h_1, h_2)$ 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 $r_t \sim \mathcal{D}(\mu(s_t, h_t))$, where $\mathcal{D}$ is any suitable distribution with mean $\mu(s_t, h_t)$. We define $h^*_t = \arg\max_{h \in \mathcal{H}} \mu(s_t, h)$ 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

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

(1)

where the expectation is taken over the randomness in the choice of $s_t$ and the reward $r_t$. 
We use a siamese transformer-based deep neural network (DNN) that takes as input a pair of queries which is unknown to the learning agent. We then compare BLIP-CTS and BBB-CTS in terms of the derivation of the update equations for both algorithms is provided in the Appendix.

3 Algorithms

We use a siamese transformer-based deep neural network (DNN) that takes as input a pair of queries (in the form of GLOVE embedding) 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 \( 1(\langle P(h_i), P(h_j) \rangle \geq \tau) \), where \( \tau \) is a pre-defined threshold, and \( 1[\cdot] \) 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 the Appendix.

Bayesian Linear Probit Contextual Thompson Sampling (BLIP-CTS) : BLIP-CTS performs bayesian generalized linear regression on top of the representation of the last layer of the transformer DNN. 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

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

where \( W_s \) is an unknown matrix, and \( \phi \) denotes the standard Gaussian CDF. Intuitively, warps the source query \( s_t \) onto the space of head queries using the matrix \( W_s \), thus learning the proper alignment between head and tail query spaces. We assume that the entries in \( W_s \) are drawn independently from a Gaussian distribution with parameters \( \mu_{ij}, \sigma_{ij} \). BLIP-CTS maintains a posterior distribution \( W_t \) over \( W_s \). At time \( t \), it samples \( W_t \sim \mathcal{N}(0, \beta^{-1}) \) 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 \( \mathcal{N}(0, \beta^{-1}) \) 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 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. In our experiments, since we assume a linear reward model as in (2), in Figure 1 that BLIP outperforms BBB.

4 Experiments

Simulations: We first compare the algorithms in a simulated setting, where the reward model follows (2). We take two sets \( S, |S| = 2500 \) and \( \mathcal{H}, |\mathcal{H}| = 500 \), where each \( x \in S \cup \mathcal{H} \) is a 3-dimensional vector generated from standard Gaussian distribution. We define a \( W_s \in \mathbb{R}^{3 \times 3} \), which is unknown to the learning agent. We then compare BLIP-CTS and BBB-CTS in terms of the regret defined in (1). The regret averaged over 10 runs is reported in Figure 1b. Since BLIP-CTS significantly outperforms BBB-CTS, we use BLIP-CTS to reformulate queries from a real dataset.

Real-World Experiments : Our dataset contains 6.2M head and 1.3M tail (source) queries, from anonymized user logs from an e-commerce website. 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
Regret | $|S| = 2500, |T| = 500, d = 3$
---|---
BLIP-CTS | Random_W
BBB-CTS |

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 $\hat{W}$.

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 $\hat{W}$.
We plan to study alternate non-linear reward models and evaluate BBB-CTS under these settings. Proving theoretical guarantees for BLIP-CTS is also an interesting research direction for the future.

References


We update the distribution of the weight matrix observation for a given data set. We sequentially iterate over the data and in each iteration, form the true posterior distribution and then approximate it with a best matching chosen parametric distribution. In BLIP-CTS, the approximating distribution over $W$ is the product of joint independent normal random variables, 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 in each iteration, form the true posterior distribution and then approximate it with a 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.

We 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

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

For $t = 1$ to $T$, do the following:

1. Denote the prior distribution over $W$ with mean and variance matrix $(\mu_t, \sigma_t^2)$ by

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

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

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

2. Find the closest independent normal approximation

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

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

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

$$F(\mu_{new}, \sigma_{new}^2) := KL(\hat{p}(W|h, s, y)||\hat{q}(W|\mu_{new}, \sigma_{new}^2))$$

$$= \int_W (\hat{p} \log \hat{p} - \hat{p} \log \hat{q})dW.$$ Given $\hat{p}$ is not a function of the new parameters, we only need to minimize the second part in the integral. According to (8),

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

(10)
The parameters for the posterior distribution should be

\[
\mu_{\text{new}}, \sigma^2_{\text{new}} = \arg \min_{\mu, \sigma^2} \left\{ \frac{1}{2} \int_W \hat{\rho} \left( \sum_i \sum_j \log 2\pi \sigma^2_{ij} + \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{(W_{ij} - \mu_{ij})^2}{\sigma^2_{ij}} \right) dW \right\}. \tag{11}
\]

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

\[
\mu_{\text{new},ij} = \int_W \hat{\rho} W_{ij} dW = \mathbb{E}_p[W_{ij}] \tag{12}
\]

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

\[
\sigma^2_{\text{new},ij} = \int_W \hat{\rho}(W_{ij} - \mu_{\text{new},ij})^2 dW = \mathbb{E}_p[(W_{ij} - \mu_{\text{new},ij})^2] \tag{13}
\]

The Hessian of \( F \) at the above mean and variance estimates comes out to be positive definite, hence satisfying the sufficient condition for them being the minimizer for \( F \).

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

\[
\frac{\partial \log Z}{\partial \mu_{ij}} = \frac{1}{Z} \frac{\partial Z}{\partial \mu_{ij}} = \frac{1}{Z} \int_W v(W) \frac{\partial q}{\partial \mu_{ij}}(W)dW = \frac{1}{Z} \int_W v(W) \left( \Pi_{m=1,m \neq i}^{d} \Pi_{n=1,n \neq j}^{d} \mathcal{N}(W_{mn}; \mu_{mn}, \sigma^2_{mn}) \right) \frac{\partial \mathcal{N}(W_{ij}; \mu_{ij}, \sigma^2_{ij})}{\partial \mu_{ij}} dW
\]

\[
= \frac{1}{Z} \int_W v(W) q(W) \left( \frac{W_{ij} - \mu_{ij}}{\sigma^2_{ij}} \right) dW = \frac{1}{\sigma^2_{ij}} (\mathbb{E}_p[W_{ij} - \mu_{ij}])
\]

Putting this in (12), we get

\[
\mu_{\text{new},ij} := \mu_{ij} + \sigma^2_{ij} \frac{\partial \log Z}{\partial \mu_{ij}}. \tag{14}
\]

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

\[
\frac{\partial \log Z}{\partial \sigma^2_{ij}} = \frac{1}{Z} \int_W v(W) \left( \Pi_{m=1,m \neq i}^{d} \Pi_{n=1,n \neq j}^{d} \mathcal{N}(W_{mn}; \mu_{mn}, \sigma^2_{mn}) \right) \frac{\partial \mathcal{N}(W_{ij}; \mu_{ij}, \sigma^2_{ij})}{\partial \sigma^2_{ij}} dW
\]

\[
= \frac{1}{Z} \int_W \hat{\rho}(W) \left( -\frac{1}{2\sigma^2_{ij}} + \frac{1}{2} \frac{(W_{ij} - \mu_{ij})^2}{\sigma^4_{ij}} \right) dW
\]

\[
= -\frac{1}{2\sigma^2_{ij}} + \frac{1}{2\sigma^4_{ij}} (\mathbb{E}_p[W_{ij}^2] - 2\mu_{\text{new},ij}\mu_{ij} + \mu^2_{ij})
\]

\[
= -\frac{1}{2\sigma^2_{ij}} + \frac{1}{2\sigma^4_{ij}} \left( \mathbb{E}_p[W_{ij}^2] - (\mu_{\text{new},ij})^2 + \sigma^4 \left( \frac{\partial \log Z}{\partial \mu_{ij}} \right)^2 \right)
\]
Since $\mathbb{E}_q[W_{ij}^2] - (\mu_{new_{ij}})^2$ is $\sigma^2_{new_{ij}}$, so the update equations are:

$$\sigma^2_{new_{ij}} = \sigma^2_{ij} + \sigma^2_{ij} \left( 2 \frac{\partial Z}{\partial \sigma^2_{ij}} - \left( \frac{\partial \log Z}{\partial \mu_{ij}} \right)^2 \right)$$  \hspace{1cm} (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 (5), then it is easy to see the re-write the above update equations as follows:

$$\delta^2 = \beta^2 + (h \circ h)^T \sigma^2 (s \circ s)$$

$$\mu_{new} = \mu + \frac{r}{\delta} \nu \left( \frac{rh^T \mu s}{\delta} \right) \left[ hs^T \circ \sigma^2 \right]$$

$$\sigma^2_{new} = \sigma^2 \left[ 1 - \frac{1}{\delta^2} \omega \left( \frac{rh^T \mu s}{\delta} \right) \left[ (hs^T \circ hs^T) \circ \sigma^2 \right] \right],$$

where $\nu(z) = \frac{N(z;0,1)}{\sqrt{\pi}}$ and $\omega(z) = \nu(z)(\nu(z) + z)$, and $\circ$ denotes the Hadamard product.

### 6.2 BBB-CTS Algorithm

In this section, we discuss the application of Bayes by Backprop algorithm [11] in the contextual bandit based query reformulation problem. We assume a posterior distribution over weighs $W$ of a neural network,

$$P(W|D) = \frac{P(D|W)P(W)}{P(D)} = \frac{P(D|W)P(W)}{\int_W P(D|W)P(W)dW},$$  \hspace{1cm} (16)

where $D$ is the training data occurring sequentially with time in the form of a set $\{(h, s, r)\}_1^t$, $P(W|D)$ is the posterior probability of $W$, $P(D|W)$ is the likelihood of $W$, $P(W)$ is the prior probability on $W$, and $P(D)$ is the evidence (also called the marginal likelihood) of the data. Furthermore, the inference is done by taking the weighted expectation over all possible values of $W$:

$$P(\hat{r}|h, s) = \mathbb{E}_{P(W|D)}[P(\hat{r}|h, s, W)] = \int P(\hat{r}|h, s, W)P(W)dW,$$  \hspace{1cm} (17)

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

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

We can now construct a cost function and compute its minimizer as our solution:

$$F(D, \theta) = \int q(W|\theta) \log \frac{q(W|\theta)}{P(W)} - q(W|\theta) \log P(D|W)dW$$  \hspace{1cm} (18)

$$= \text{KL}[q(W|\theta)\|P(W)] - \mathbb{E}_{q(W|\theta)}[\log P(D|W)].$$  \hspace{1cm} (19)

Calculating the expectation of the likelihood over the variational posterior is computationally intractable, so we approximate it by a tractable cost function using sampled weights as follows:

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

We can now use automatic differentiation as provided by frameworks such as PyTorch. We only look into the sampling of weights and setting up the cost function as above. We can then leverage the
### Algorithm 2: BBB-CTS

1. **Input:** Parameters $\beta > 0$, $\mu_0, \rho_0 \in \mathbb{R}^{d \times 1}$, learning rate $\alpha$
2. **For** $t = 0, 2, \cdots, T$ **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 $s_t \in S$
   7. Choose optimal action $h_t = \arg \max_{k \in \mathbb{M}} h_t^T W_t s_t$ (following the reward in (2)).
   8. Observe reward by sampling $r_t \sim \phi \left( h_t^T W_t s_t \right) / \beta$
   9. Let $f(W_t, \theta_t) = \log q(W_t | \theta_t) - \log p(W_t) p(D | W_t)$, where $D = \{ h_k, s_k, r_k \}_{k=0}^T$
   10. Calculate the gradient with respect to the mean $\mu$: $\nabla \mu = \frac{\partial f(W, \theta)}{\partial \mu} + \frac{\partial f(W, \theta)}{\partial \rho}$.  
   11. Calculate the gradient with respect to the standard deviation parameter $\rho$: $\nabla \rho = \frac{\partial f(W, \theta)}{\partial \rho} \cdot \epsilon / (1 + \exp(-\rho)) + \frac{\partial f(W, \theta)}{\partial \rho}$.  
   12. Update the variational parameters: $\mu \leftarrow \mu - \alpha \nabla \mu$, $\rho \leftarrow \rho - \alpha \nabla \rho$.

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

### 6.3 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 $M_0$.

<table>
<thead>
<tr>
<th>$M_0$</th>
<th>$M_1$ (BLIP-CTS)</th>
<th>$M_0$</th>
<th>$M_1$ (BLIP-CTS)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Source 1:</strong> under armour underw</td>
<td><strong>Source 2:</strong> keter outdoor trash can trash bags</td>
<td></td>
<td></td>
</tr>
<tr>
<td>under armour wwp</td>
<td>under armour tech boxerock</td>
<td>hefty step garbage can</td>
<td>outdoor trash bags</td>
</tr>
<tr>
<td>under armour barcen</td>
<td>under armour magneto pro</td>
<td>outdoor trash can storage shed</td>
<td>black outdoor trash bags</td>
</tr>
<tr>
<td>under armour sportstyle</td>
<td>under armour tech</td>
<td>outside trash bin</td>
<td>hefty step garbage can</td>
</tr>
<tr>
<td>under armour culver</td>
<td>under armour tech 2.0</td>
<td>outside trash bin storage</td>
<td>large outdoor trash bags</td>
</tr>
<tr>
<td>under armour tech 2.0</td>
<td>under armour tech 2.0</td>
<td>outside trash can storage</td>
<td>grow bags 30 gallon with handles</td>
</tr>
<tr>
<td><strong>Source 3:</strong> auto shade, land rover discovery</td>
<td><strong>Source 4:</strong> rug dig bed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>las vegas sail boat</td>
<td>dodgers sunshade for cars</td>
<td>rug foam</td>
<td>under bed rug</td>
</tr>
<tr>
<td>shade shore</td>
<td>car cover outside land rover l3</td>
<td>bed rug</td>
<td>rug for under bed</td>
</tr>
<tr>
<td>a shade of vampire 77</td>
<td>carolina skiff boat cover</td>
<td>rug for bed</td>
<td>bed rug</td>
</tr>
<tr>
<td><strong>Source 5:</strong> tye die use shirt</td>
<td><strong>Source 6:</strong> cat 6a 321t</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tye die tshirt</td>
<td>tye die tshirt</td>
<td>cat 6a</td>
<td>cat 6a 2000</td>
</tr>
<tr>
<td>tye die shirt</td>
<td>tye die tshirt</td>
<td>cat 6a plenum</td>
<td>cat 6a</td>
</tr>
<tr>
<td>state line tack</td>
<td>tie die shirt kit</td>
<td>cat 6a plemum</td>
<td>cat 6a 5000</td>
</tr>
<tr>
<td>tye die shirt kit</td>
<td>state line tack</td>
<td>fdi tl-232r-3v3</td>
<td>cat 6a 10000</td>
</tr>
<tr>
<td><strong>Source 7:</strong> oneplus 5 sticker skin</td>
<td><strong>Source 8:</strong> 2006 prius rubber mats</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gameboy sticker</td>
<td>gameboy sticker</td>
<td>toyota corolla rubber floor mats</td>
<td>dodge ram rubber floor mats</td>
</tr>
<tr>
<td>lightening mcqueen stickers</td>
<td>airpod sticker skin</td>
<td>2002-1250 floor mats</td>
<td>toyota corolla rubber floor mats</td>
</tr>
<tr>
<td>final fantasy 7 decal</td>
<td>lightening mcqueen stickers</td>
<td>2007-1150 floor mats</td>
<td>toyota tacoma rubber floor mats</td>
</tr>
<tr>
<td>paint by number anime stickers 400 pcs</td>
<td>iphone x sticker skin</td>
<td>2009 canny floor mats</td>
<td>honda accord rubber floor mats</td>
</tr>
<tr>
<td><strong>Source 9:</strong> atoms sound bar</td>
<td><strong>Source 10:</strong> dark green twill tape</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gogroove sound bar</td>
<td>megacra sound bar</td>
<td>dark green tape</td>
<td>dark green tape</td>
</tr>
<tr>
<td>nakamichi sound bar</td>
<td>wetsound sound bar skin</td>
<td>luminous tape</td>
<td>dark brown tape</td>
</tr>
<tr>
<td>wetsounds sound bar</td>
<td>kuryakyn sound bar</td>
<td>od green tape</td>
<td>dark brown duck tape</td>
</tr>
<tr>
<td>boes sound bar</td>
<td>meidong sound bar</td>
<td>blue hockey tape</td>
<td>dark tape</td>
</tr>
<tr>
<td>naxa sound bar</td>
<td>zvox sound bar</td>
<td>florist tape green</td>
<td>od green tape</td>
</tr>
</tbody>
</table>

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.