Automating Relevance Banding in eCommerce Search using Click Model

ABSTRACT

eCommerce is burgeoning: in the past five years, both in the USA and the UK, eCommerce retail sales having overtaken brick-and-mortar stores for the first time. Search is a primary means for users engaging in eCommerce. eCommerce companies often perform laborious human-intensive mappings of queries to various categories in their product taxonomies—a process called "relevance banding." This is done in order to improve recall and precision of their search engines. In this paper, we propose fully automated alternatives to this manual process. We use statistical properties of the click-model that is constructed using query-click logs in order to automate this process. We propose two algorithms—probability banding, and entropy banding—that perform banding in a fully automated manner. In large-scale A/B testing, our algorithms demonstrate considerable revenue and orders increase over the manual banding baseline. Our algorithms are now deployed at scale at CorpX—a multibillion dollar eCommerce major.

1 INTRODUCTION

When a user submits a query to an eCommerce search engine, the search engine tries to understand the intent behind the query, and subsequently selects from its inventory those items that match the query intent. In order to specify the query intent, the most important queries (by frequency of occurrence, and monetization) are *mapped* to sets of nodes of the product taxonomy. A taxonomy is a DAG of many nodes (also called *categories*): a main-category has children categories, each of whom further has children subcategories, as shown in Fig. 1. Each product in the inventory is placed into one or more categories in the taxonomy. If present, such a taxonomy provides a human-curated layer of abstraction and classification over the actual items. Since the sets of categories mapped to a query are also known as the *relevance bands* of the query, this process of mapping queries to sets of relevant taxonomy nodes is called *relevance banding*. *The rest of the paper focuses on eCommerce search engines having a product taxonomy.*

At most eCommerce companies, including ours[1], it is a prevalent practice to *manually* perform relevance banding. This manual banding is typically performed and maintained by expert search analysts, over a period of time. Clearly this is a time-consuming process, prone to human error, and subject to an intensive amount of human bookkeeping and tracking. This leads to our problem statement:

> Can we automate relevance banding using statistical methods?

We approach this problem by using the statistical properties of *click model*, a well researched technique in the domain of *document* retrieval. Click models mine the query-click logs to find the query-document associations.

[1]A multi-billion dollar Fortune-100 company that we will call CorpX

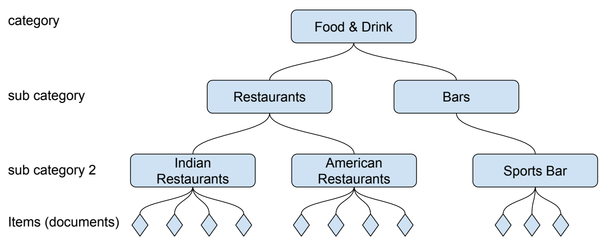

Figure 1: A part of DAG of the taxonomy used at CorpX. Since our work pertains to eCommerce, we compute the click model on sub-category-2 using items, as opposed to documents for web retrieval. Details in text.

In this paper, we propose two algorithms based on click model, to automate relevance banding: *probability banding*, and *entropy banding*. Not only do these algorithms fully automate this process of relevance banding, they also outperform manual banding in search efficacy. Our algorithms demonstrate considerable revenue and order increases over the manual banding baseline in large-scale A/B testing.

2 RELATED WORK

In web search domain, the problem of associating queries to a set of categories has been studied in detail. Queries can be classified along multiple dimensions, including *search goals* [2, 9], *location-sensitivity* [13], and *semantic topics* [10, 12]. Of these, the most closely related works are on *topical query classification.* Authors in [12] train a bridging classifier on an intermediate web taxonomy, and tackle the sparseness of query features by augmenting queries with search engine results. In [10], instead of improving feature representation, authors chose to increase the amounts of training data by semi-supervised learning with click-graphs. However, there is scant literature on classifying queries into the product taxonomy nodes in eCommerce domain, and its subsequent application to search relevance. One key difference between web documents and ecommerce items is that latter are *pre-mapped* into the target categories as opposed to former.

Similarly, there is extensive literature on click models and their applications to web search. We refer the reader to an excellent survey [4] on the increasingly rich landscape of click models. In recent years, many click models have been proposed and used in various ways, including towards *improving document ranking in web-search* [1, 6] and towards *evaluating web search results* [5]. In the context of eCommerce, authors in [11] discuss an application of click model for *atypical query identification.* However, to the best of our knowledge, there is no study on applying click models to eCommerce for search relevance. The presence of product taxonomies in eCommerce search engines means that standard click model techniques from document retrieval do not carry over to this domain. In this paper, we address this gap in literature, applicable to a domain of high commercial importance, by applying the click model to automate relevance banding for eCommerce.

3 BACKGROUND

In this section, we elaborate on the concepts used in our algorithms.

3.1 Click Model

In the context of a taxonomy, when a user makes a query, and then clicks on items in the result set, these clicks are recorded against the categories that the items are in. For each query, we aggregate these user clicks over a certain time period. These aggregated clicks, when normalized, form a probabilistic distribution over categories. This distribution of clicks that results from a query are formalized into the *click model* [4].

Definition 3.1 (Click Model). The probabilistic distribution of association of a query $Q_i \in \mathcal{Q}$ to the taxonomy $\mathcal{T}$ is denoted by $\Pr_{Q_i}(\mathcal{T})$, and is defined as

$$\Pr_{Q_i}(\mathcal{T} = C_j) = \frac{clicks_{Q_i,C_j}}{\sum_{j=1}^{j=m} clicks_{Q_i,C_j}}$$

Where $clicks_{Q_i,C_j}$ denotes the number of user clicks for query Q_i pertaining to the items in result-set belonging to category[2] C_j.

Although clicks provide an important source of implicit feedback, and have been shown to be an effective method to extract relevance information, research shows that clicks suffer from a number of biases. In the following paragraphs, we explain the biases and the corrective measures we have employed.

3.1.1 Appearance Bias Correction. When presented with the search results, a user tends to click on a result based on the accompanying meta-data or abstract. Therefore, a result may spuriously trick the user into clicking the result. We use *whether the clicked item was purchased* as a proxy to determine the importance of a click. Accordingly,

$$\Pr_{Q_i}(\mathcal{T} = C_j) = \frac{clicks_{Q_i,C_j} + \alpha \times purchases_{Q_i,C_j}}{\sum_{j=1}^{j=m}(clicks_{Q_i,C_j} + \alpha \times purchases_{Q_i,C_j})}$$

Where $purchases_{Q_i,C_j}$ denotes the *number of aggregated purchases* on query Q_i pertaining to the items belonging to category C_j.

3.1.2 Position Bias Correction. Studies show that a user is less likely to click on a result at a lower position in the result set, albeit being relevant [8]. To correct for this bias, we process each click/order activity with a position correction factor β applied. Such a function is application specific; we use the following function in our system, starting at position $p = 1$:

$$\beta(p) = 1 + \frac{\log(min(p, \mathcal{P}))}{\log(\mathcal{P})}$$

Where $\mathcal{P}$ indicates the position after which $\beta(p)$ becomes constant. We empirically tune the value of $\mathcal{P}$ to be 30 for our system.

Therefore, if A is the actual activity performed by a user at position p, then position corrected activity A_p is $A_p = \beta(p) \times A$.

[2]In this formalization, we have used the generic term "category" to mean a category, a sub-category-1, or a sub-category-2 in the taxonomy. In practise, our click model is computed for sub-category-2 level nodes.

3.1.3 Temporal Bias Correction. User interests and behaviors in e-commerce may change over time [3]. Accordingly, we wish to stress recent activity (clicks, purchases, etc.) while mining query to category associations, and de-stress past activity. Therefore, each activity is decayed with a daily time decay factor $\theta \in (0, 1]$. This also enables the click model to adapt to changes in the taxonomy.

If $A(t)$ is the actual activity performed by a user at time t, and t_f is the time when click model is constructed, then the time decayed count of activity $A(t_f)$ is $A(t_f) = A(t) \times \theta^{(t_f - t)}$.

3.2 Information Theory

The (information theoretic) entropy $H(X)$ of a discrete random variable X is defined by:

$$H(X) = -\sum p(x) \log p(x)$$

4 ALGORITHMS

We now gather all the components defined in the previous section to compute the relevance bands for a given query Q_i . We propose two algorithms based on the click model query-category probability distribution: probability banding, and entropy banding.

4.1 Probability Banding

Idea and Algorithm. The probability banding algorithm uses the click model probability distribution $\Pr_{Q_i}(\mathcal{T})$ for Q_i to place the most important categories from the taxonomy $\mathcal{T}$ associated to Q_i into a *single* relevance band.

Algorithm 1 ProbabilityBanding (max_categories, min_prob)

```
Require: sorted_category_list : list of categories sorted in decreasing order of probability given by $\Pr_{Q_i}(\mathcal{T})$
 1: band ← { };                                  ▷ Initializing the band
 2: for Each category $C_j$ in sorted_category_list do
 3:     if (band.size() ≥ max_categories) then
 4:         break;
 5:     end if
 6:     if ($\Pr_{Q_i}\{\mathcal{T} = C_j\} \leq min\_prob$) then
 7:         break;
 8:     end if
 9:     band.add($C_j$);
10: end for
11: return band;
```

We walk the reader through the probability banding algorithm. First, it obtains the list of categories sorted in decreasing order of the probability given by $\Pr_{Q_i}(\mathcal{T})$. Then, it iterates over the sorted list of categories, adding the categories into the band until any of the following happens:

- the number of categories added reach the maximum number of categories allowed for the band (lines 3-5)
- hits a category with probability less than the minimum threshold (lines 6-8)

Finally, it returns the single relevance band it has constructed, comprising the most important categories for Q_i .

Determination of parameters.

- *max_categories*: Empirically, we determine that on an average, the probability of top four categories sums up to 0.9 for the queries in our database. Accordingly, we set the maximum number of categories at 4.
- *min_prob*: We chose a value of 0.02 since a lower probability than that cannot be justified to be statistically significant, and is deemed to be noise.

4.2 Entropy Banding

Probability banding considers a category with a high probability and a category with relatively lower probability equally important, by placing them in the same band. For example, in a scenario where a query, say, *pizza* is associated with the category *pizza* with a probability of 0.70, and with the category *italian-restaurants* with a probability of 0.25, probability banding will place both the categories in one band even though the given distribution implies category *pizza* is significantly more associated with query *pizza* than category *italian-restaurants*. This leads us to the conclusions that the probability distribution should be interpreted in a more sophisticated manner to generate a *list of bands* than merely to filter the most important categories.

There are, therefore, two questions we must answer: how many bands should be created, and what should be the "cut-offs" for each band? We build an algorithm around information theoretic entropy to answer both.

Intuitively, we agree that categories with similar probability should reside in one band. Let us make precise the notion of similarity. *If the probability of a new category deviates more than a certain proportion of the average probability of categories in the band, that new category will be considered dissimilar to the ones in the band.*

It remains to define the proportion used in the definition above. We use information theoretic entropy in order to define this proportion. We explain this in the determination of parameters later in this section.

With this idea, we describe our algorithm for entropy banding for a given query Q_i and its probabilistic distribution over categories as $\Pr_{Q_i}(\mathcal{T})$ in Algorithm 2.

We walk the reader through the entropy banding algorithm. First, it obtains the list of categories sorted in decreasing order of the probability given by $\Pr_{Q_i}(\mathcal{T})$. It then iterates over the sorted list of categories. If the probability of the next category significantly changes w.r.t. to the current band, a new band is created (lines 6-10).

Determination of parameter λ. The deviation factor λ represents the proportion of average probability of the band, by which the next category is allowed to deviate, in order to still be considered similar to the existing categories in the band. Intuitively, a large value of λ implies that a broader range of probabilities can be part of a single band.

Turning this reasoning around, we see that for those queries whose probability distribution is concentrated, λ should be large; conversely, for those queries whose distribution is spread more evenly over many categories, λ should be small.

Algorithm 2 EntropyBanding(λ)

```
Require: sorted_category_list : list of categories sorted in decreasing order of probability given by $\Pr_{Q_i}(\mathcal{T})$
 1: band_prob_avg ← 0.0;
 2: previous_prob ← 0.0;
 3: band_list ← [ ];
 4: current_band ← { };
 5: for Each category $C_j$ in sorted_category_list do
 6:     if (previous_prob − $p(C_j)$) > $\lambda \times$ band_prob_avg then
 7:         band_list.add(current_band);
 8:         current_band = { };
 9:         previous_prob = 0.0;
10:     end if
11:     current_band.add($C_j$);
12:     previous_prob = $p(C_j)$;
13:     update band_prob_avg;
14: end for
15: return band_list;
```

We know that entropy of a random variable quantifies exactly the concentration of its probability distribution. Therefore, to model the above explained behavior, we define λ for a query Q_i as:

$$\lambda_{Q_i} = 2^{-H(\Pr_{Q_i}(\mathcal{T}))}$$

5 EMPIRICAL VALIDATION

We performed two experiments to assess the efficacy of our algorithms. In the first experiment, we compare the probability banding algorithm against the baseline manual banding. In the second experiment, we compare the entropy banding algorithm against the probability banding algorithm.

5.1 Click Model Dataset

To train the click model, we use the click and purchase activities on *CorpX* search engine during the period of *Dec 1, 2014 to May 30, 2016*. Our query space consists of 148,737 queries. These queries were issued 101,379,388 times by 36,598,660 users in the given time period. Our inventory consists of 1,927,349 items placed into 1,517 categories in the taxonomy. For the aforementioned queries, there are 210,754,810 clicks and 16,727,410 purchases distributed over the categories in the product taxonomy. While training the click model, we discard the query-category pairs with less than 30 clicks so as to reduce noise. After training, we obtain 863,403 query-category pairs with probability more than 0.02.

5.2 Experiment 1: Probability Banding vs. Manual Banding

In this experiment, we wish to compare the probability banding algorithm against the manual banding.

To compare the two methods, we use an A/B testing framework. Our baseline comprises of manual relevance bands developed by a team of expert search analysts at *CorpX* over a period of 6 months, and our variant has relevance bands computed through probability banding algorithm. We diverted equal amount of search requests to probability banding and baseline manual banding for a period of *15 days* from *June 24, 2016 to July 10, 2016*. During this time period, set of 148,737 queries were issued through 11,718,958

Table 1: A/B test results

	Net Revenue Increase	Orders Increase
Experiment 1	+3.366%	+2.922%
Experiment 2	+1.087%	+0.193%

sessions (5, 861, 895 sessions with manual banding and 5, 857, 063 sessions with probability banding), and the inventory comprised of 1, 089, 833 items.

We observed a statistically significant increase of +3.366% in net revenues and an increase of +2.922% in orders. Results are summarized in Table 1.

5.3 Experiment 2: Entropy Banding vs. Probability Banding

In this experiment, we wish to compare the entropy banding algorithm against the probability banding algorithm.

To compare the two methods, we use an A/B testing framework. Our baseline comprises of relevance bands computed through probability banding algorithm, and our variant has relevance bands computed through entropy banding algorithm. Again, we diverted equal amount of search requests to entropy banding and probability banding during the period of *September 21, 2016 to October 9, 2016*. During this time period, set of 148, 737 queries were issued through 6, 654, 113 sessions (3, 321, 295 sessions with probability banding & 3, 332, 818 sessions with entropy banding).

We observed a statistically significant increase of +1.087% in net revenues and an increase of +0.193% in orders. Results are summarized in Table 1.

5.4 Discussion

Our broad hypothesis was that automated methods, based on click model, are not only more efficient operationally, but can outperform manual banding as far as search efficacy. This is proved by the significantly positive results of both our experiments. Readers relatively unfamiliar to A/B testing in eCommerce should kindly note that increases of 0.5% are considered significant improvements; whereas our revenue increases are 3.3% and 1.1% for the two experiments.

Furthermore, the entire click model banding process is fully automated, requiring no human intervention at any stage. This contrasts with the intensive manual mapping and re-mapping (upon seeing search results) done previously.

Let us now compare the two experiments. Note that when the entropy of a query's click model distribution is low, the first band will likely have all the significant categories. In that case, there is not much difference between the two algorithms. On the other hand, when the entropy of a query is high, we would expect entropy banding to make a difference. As it stands, queries with high entropy (≥ 2) are a significant minority (8% of total 148, 737 queries). It is on these smaller set of queries that entropy banding will generate an increase in revenue. This is borne out by the result in Table 1.

6 CONCLUSION

We began our work with the hypothesis that the intensive and error-prone process of manual banding can be improved using automated statistical methods. We proposed two algorithms: probability banding, and entropy banding, in order to accomplish this automation. Both of them rely on the click model.

Probability banding is conceptually simple, and generates a single band of highly associated categories. Entropy banding aims to outperform probability banding on high-entropy queries. It generates (potentially) multiple bands, using information theoretic criteria to determine the band boundaries. Our A/B tests show significant increases in revenue and order generation over manual banding as a result of both these algorithms.

Our system is now deployed at scale at CorpX—a multi-billion dollar company—, and further innovations based upon it are being experimented with. Although we are not at liberty to disclose precise revenue increase figures, they are in the millions of dollars (per financial quarter) range.

Future work points to algorithms that segment queries based on other statistical properties (including ambiguity), and tailoring of algorithms towards each segment. We are currently experimenting with a combination of entropy, frequency, and monetization. Another line of future work is to use these algorithms on query categorization using knowledge resources [7, 12].

REFERENCES

[1] Alexey Borisov, Ilya Markov, Maarten de Rijke, and Pavel Serdyukov. 2016. A Neural Click Model for Web Search. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 531–541.

[2] Andrei Broder. 2002. A taxonomy of web search. In *ACM Sigir forum*, Vol. 36. ACM, 3–10.

[3] Mu-Chen Chen, Ai-Lun Chiu, and Hsu-Hwa Chang. 2005. Mining changes in customer behavior in retail marketing. *Expert Systems with Applications* 28, 4 (2005), 773–781.

[4] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. Click Models for Web Search. *Synthesis Lectures on Information Concepts, Retrieval, and Services* 7, 3 (2015), 1–115.

[5] Aleksandr Chuklin, Pavel Serdyukov, and Maarten De Rijke. 2013. Click model-based information retrieval metrics. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 493–502.

[6] Georges Dupret and Ciya Liao. 2010. A model to estimate intrinsic document relevance from the clickthrough logs of a web search engine. In *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 181–190.

[7] Laura Hollink, Peter Mika, and Roi Blanco. 2013. Web usage mining with semantic analysis. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 561–570.

[8] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. 2007. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)* 25, 2 (2007), 7.

[9] Uichin Lee, Zhenyu Liu, and Junghoo Cho. 2005. Automatic identification of user goals in web search. In *Proceedings of the 14th international conference on World Wide Web*. ACM, 391–400.

[10] Xiao Li, Ye-Yi Wang, and Alex Acero. 2008. Learning query intent from regularized click graphs. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 339–346.

[11] Neeraj Pradhan, Vinay Deolalikar, and Kang Li. 2015. Atypical Queries in eCommerce. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 1767–1770.

[12] 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.

[13] Xing Yi, Hema Raghavan, and Chris Leggetter. 2009. Discovering users' specific geo intention in web search. In *Proceedings of the 18th international conference on World wide web*. ACM, 481–490.