Label aggregation approaches for Learning to Rank in E-commerce

ABSTRACT

We explore different label aggregation approaches to train a one global Learning To Rank (LeToR) [10] model that learns from multiple label sources. These datasets are typically available in a large scale E-commerce organization like Walmart. LeToR models trained on click logs are somewhat self-fulling in nature [2]. Label aggregation approaches try to reduce the multi-objective LeToR optimization problem to a single objective. As opposed to model aggregation approaches where one aims to train models on each label source and combine at the end, label aggregation approaches have the advantage of being easy to design, maintain, iterate on and are faster to take to production. Datasets are derived from multiple sources Query-Product annotations, Query annotations and Customer Engagement. There are multiple approaches to label aggregation. In this paper, we discuss the different datasets, how we can leverage them to build LeToR models, and how we can evaluate the different LeToR models. The best set of models are ones that performs optimally across different evaluation metric(s) and lie on the Pareto Frontier [12]. We outline steps to come up with such models.

KEYWORDS

Ranking, LeToR, Multi-Objective, Experimentation

ACM Reference Format:

. 2021. Label aggregation approaches for Learning to Rank in E-commerce. In *Proceedings of Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Product search is an important tool in any E-commerce website, users issue a search Query and want to see the related Products. They perform some action, like clicking on Products, adding Products to the cart or buying the Product. Aim of the search algorithm is to show relevant products to the users, at the same time the products should be of good quality and should make the users engage with it. The search algorithm has to take care of multiple aspects such as Engagement, Relevance, revenue generation, freshness etc. The general practice of training LeToR models on click logs are somewhat self-fulling [15] in nature [2]. Using multiple label sources on the other hand allows the model to generalize and directly aim to optimize for different goals/aspects.

In a large E-commerce organization like Walmart there exist click stream logs that capture user actions on a search page. For a Product in the results, the user may choose to do nothing, click, add to cart (atc) or order the Product. [8] discusses clicks, add to cart and order signals as multiple feedback signals. We propose ways to combine these signals to generate a single label.

In our experience, we have found that feedback from Web users and App users are different in nature too. App users seem to prefer different products for certain queries. In light of this, we explore different label aggregation approaches to combine feedback from Web and App. We combine feedback from Web and App channels using Inverse Propensity Weighting [17] in one novel approach. In addition to this, there exists tools that allow merchants or human judges to make Query-Product annotations or Query annotations. A human judge evaluates a Product for relevance with a given query, this is an example of Query-Product annotations. A merchant declaring that we should not be displaying "refurbished tvs" in response to a Query like "tvs" is an example of a Query annotation. In addition to this, there exists external teams like SEO that help us with information on how a Product has performed over the SEO keywords. These are useful feedback signals that needs to be incorporated in the LeToR modelling process. Having several sources can compensate for weakness in a single source [19]. In Section 3.2 we talk about the different datasets that we use to train the LeToR model.

Several studies discuss solutions to the multi-objective optimization problem in the context of Product search [5, 11, 13, 14]. The solution can be classified into 2 categories.

1. Solutions where multiple models are trained, one for each label source. The results from the multiple models are then aggregated in some way to come up with a final model score that is used for ranking. These approaches are classified as **model aggregation** approaches.

2. Solutions where the multi-objective problem is reduced to a single objective problem in order to build a one global model. These approaches are classified as **label aggregation** approaches.

Label aggregation approaches as opposed to Model aggregation approaches are easier to design, maintain, iterate on and are faster to take to production. From a practical point of view, we favor working with one global model instead of training multiple models. In Section 3.2 we discuss several label aggregation approaches.

To evaluate the different models, we defined as benchmark the NDCG on the test set for each label source as an objective. As discussed in [5] non-trivial multi-objective problems typically do not have a feasible solution that maximizes all objective at simultaneously. Therefore, we look for Pareto [12] optimal solutions; that is, solutions that cannot be improved for one objective without degrading at least one of the other objectives. The set of Pareto optimal solutions is called the Pareto frontier [12]. The optimality

SIGIR '21, July, 2021,

© 2021 Association for Computing Machinery.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00

https://doi.org/10.1145/1122445.1122456

expectation is that an optimal solution for the resulting single-objective optimization problem would be on the Pareto Frontier of the multi-objective optimization problem.

We take an experimental approach, where we discuss ways to combine datasets from multiple label sources. We iterate over different ways of combining each type of label source. The model that performs optimally given the evaluation approach is one where we combined App and Web datasets through Max-Rank/Stochastic aggregation approach and we used stochastic aggregation approach to combine labels from Query Product annotations and also labels from Query Annotation. In Section 4 we talk about the experimental setup and experiments in detail. We list out all models that lie on the Pareto frontier.

2 RELATED WORK

Multiple studies have investigated the relation between Engagement signals and Relevance signals in Product Search for E-commerce [1, 8, 9, 11, 16]. [8] show the ineffectiveness of just using relevance ratings for Product search and discuss click, add to cart and order rate as multiple source signals to train LeToR models. As far as we know, this is the first work that looks at App and Web channels as separate labelling sources. [3] talks about the different aspects that a user cares about before making a product purchase making Product search a challenging problem. We consider other Query annotation and Query Product annotation datasets that are available in an E-commerce organization. These datasets capture aspects important for our customers.

Multiple-objective optimization problem in the context of Product search is a widely studied problem [5, 13, 14, 18]. [5, 19] focus on approaches to combine Relevance and Engagement signals through Label aggregation approaches. [19] define a tiered ranking approach where they treat Relevance labels as top tier and treat labels derived from clicks as a second tier. In practice, Relevance labels are far fewer in number as compared to Engagement labels. This approach is not defined for cases where Query Product pairs do not have Relevance labels. [5] combine Relevance labels and Engagement labels using a stochastic aggregation approach, where they sample some Queries and use only Relevance labels for these Queries. For the other set, they use Engagement labels. Under certain conditions they show that this approach leads to Pareto optimal solutions. We use stochastic approach to combine Engagement labels with labels from Query annotation dataset and labels from Query Product annotation dataset. For the different Engagement labels (click-rate, add to cart rate, order-rate) and also datasets from Web and from App, we implement multiple such approaches in a method that takes in all the different signals, combines them and returns a final label. We experiment with multiple ideas, inverse propensity weighting [17] typically used in E-commerce search to debias clicks [7] seems to give us good results but max rank strategy albiet simple lie on the Pareto frontier.

We adopt the evaluation approach used in [5] to evaluate the different models. We define NDCG [6] on each of the label source (Query annotation, Query-Product annotation, Web engagement, App engagement). The optimal model that lies on the Pareto frontier is one that uses all the sources while training the model.

3 LETOR IN E-COMMERCE

3.1 Problem Setting

We have given a set of queries $Q = \{q_1, q_2, \ldots, q_n\}$. For each Query $q \in Q$ there is set of Products $P_q = \{p_1, p_2, \ldots, p_k\}$ that are related to Query q. The pair (q, p_i) will have a real value label $l(q, p_i)$, higher the value of label $l(q, p_i)$, indicates p_i is more related to query q. For a Query q, a label set is represented as $L_q = \{l(q, p_1), l(q, p_2), \ldots, l(q, p_k)\}$. (q, P_q, L_q) is one training instance for the algorithm to learn. The algorithm aims learns to rank the products P_q for a given Query q such that the ranking produced by algorithm have a maximal agreement with L_q which is quantified using NDCG [19] metric.

3.2 Multiple label sources

There are multiple label sources that capture different aspects important for the customers. Ideally, the LeToR model needs to optimize for all of these objectives simultaneously. Let's say we have two label sources l_1 and l_2 and cost related to them will be computed using $L_{q1} = \{l_1(q, p_1), l_1(q, p_2), \ldots, l_1(q, p_k)\}$ and $L_{q2} = \{l_2(q, p_1), l_2(q, p_2), \ldots, l_2(q, p_k)\}$. The combined cost will be $cost = (cost_{l1}, cost_{l2})$. One $cost$ is said to be smaller than the other cost when either, costs related to both the label sources are smaller or cost with respect to one label source is smaller than the other cost and costs with respect to other label sources are same.

In this section, we discuss the different datasets and multiple label aggregation approaches to combine them.

- Engagement dataset: There are two sources of engagement, dataset from Web and dataset from App. Both the label sources have similar structure and label is computed using clicks, add to carts and orders for a given Query-Product pair. We experiment with multiple approaches to get the label from click-rate, add to cart rate and order-rate.
  1. **Weighted aggregation:** In this method we combine the signals linearly by multiplying each signal by their importance.
  $final\text{-}score = w_c * click\text{-}rate + w_a * atc\text{-}rate + w_o * order\text{-}rate$, where w_c, w_a, w_o are weights which can be estimated by observing the ratio of click:atc:order rates in the engagement logs for relevant items. The exact weights are computed by dividing maximum of the event counts by a particular event count. For eg. $w_c = max(clicks, atcs, orders)/clicks$. The $final\text{-}score$ is used to rank the products.
  2. **Lexicographic aggregation:** Treat order-rate, atc-rate, click-rate as tuples and rank the products based on lexicographic ordering.
  where $(o_1, a_1, c_1) > (o_2, a_2, c_2)\ if\ o_1 > o_2\ or\ if\ o_1 = o_2\ and\ a_1 > a_2\ or\ if\ o_1 = a_2\ and\ a_1 = a_2\ and\ c_1 > c_2$.

  Web and App datasets exhibit different patterns. To combine labels from Web and App, we use
  1. **Inverse propensity weighting:** where we scale up App customer engagement in order to combine App and Web metrics. For example- $click\text{-}rate\text{-}combined = click\text{-}rate\text{-}web + \alpha_1 * click\text{-}rate\text{-}app$. α_1 is computed by considering the ratio of $click\text{-}rate\text{-}web$ and $click\text{-}rate\text{-}app$.

2. **Max-Rank:** In this method, for a given query we rank the items using Web and App signals separately. We combine this ranking by taking maximum value of ranking from Web and App. $final\text{-}rank = max(web\text{-}rank, app\text{-}rank)$
3. **Stochastic Label aggregation:** In this approach, for each Query, toss an unbiased coin with probability of head α. If head comes, take labels from one source for all the Products, else take labels from other label source. If the Query is present in only one of the source then we take the labels from that source. On an average $\alpha * |NumberOfQueries|$ will have labels from one source and $(1 - \alpha) * |NumberOfQueries|$ will have labels from other source.

- Query-Product annotation: Datasets that are derived for Query-Product pairs capture aspects other than engagement. Relevance labels obtained from a crowd is an example of this dataset. Datasets generated by a group of merchants for the purpose of campaigns or datasets retried from SEO or affiliate teams are other examples of such a dataset. We experiment with **stochastic** approach described earlier to combine such datasets with other labels. To simplify the setup for our experiments we combine all such Query-Product annotation dataset into a single relevance dataset by pre-processing and mapping each annotated dataset into relevance labels.
- Query annotations: To be able to provide a better customer experience to the users merchants annotate Query with a set of desirable Product attributes. For example, not showing "refurbished tvs" to a customer looking for "tvs". We map merchant annotations to Products to create a Query-Product labels using the Query annotations as a source. We experiment with **stochastic** approach described earlier to combine this dataset with other labels.

4 RESULTS

4.1 Datasets

As discussing in 3.2 there are 3 types of label sources.

- **Engagement datasets:** We have engagement information for $630K$ unique Queries and $26M$ unique Query-Product pairs from the Web and $500K$ unique Queries and $23M$ unique Query-Product pairs from App.
- **Query Product Annotation datasets:** Combining the different datasets that have Query-Product labels we have information about $290K$ unique Queries and $2.8M$ unique Query-Product pairs.
- **Query annotation datasets:** This label sources has $24K$ unique Queries and $2M$ Query-Product pairs.

4.2 Experiment Setup

The training dataset is divided into 80% train, 10% validation and 10% test set. The algorithm aims to optimize NDCG@10 [19] on train dataset and optimal hyper-parameters are found using validation dataset. After training the model we are calculating NDCG@10 on the full datasets to report final NDCG values. Engagement **Web-NDCG** is NDCG@10 on Web Engagement dataset, Engagement **App-NDCG** is on App engagement dataset. **QIA-NDCG** is NDCG calculated on Query-Product annotation dataset.

Features are selected based on its coverage on a 10% sub-sample of queries from whole train dataset. Models are trained using XGBoost[4] algorithm with a tree depth of 3,5,7 and min child weight from 0 to 60 with a step of 10. The regularization parameters are α and λ with both have values from (0.01, 0.1, 1). Hyper-parameters are optimized using grid search method.

4.3 Experiment results

The experiments are divided into four parts, first a baseline model to benchmark. Subsequently, in each phase a new dataset is introduced and results are compared with baseline (benchmark).

4.4 Baseline Model

Baseline model is trained using only Engagement from Web. We experiment with **Lexicographic aggregation** and **Weighted aggregation** approaches to combine clicks, atcs and orders signals. The NDCG values are written in table 1. **Lexicographic aggregation** performs better than Weighted aggregation. In the following sections, we use Lexicographic aggregation approach to combine clicks, atcs and orders.

Method name	**Web NDCG**	**App NDCG**	**QIA NDCG**
Lexicographic aggregation	**1**	**1**	**1**
Weighted aggregation	-0.44%	-0.46%	0.51%

Table 1: Baseline models. Results using only Web Engagement dataset

Table 2 compares results of various model combinations with the baseline model. Experiments are color coded according to the label data we are combining with the Web label data, experiments in light green adds App label dataset, experiments in light blue adds App label dataset and Query-Product annotation dataset and experiments in brown color uses all the three datasets. The percentage increase in NDCG's is in bold for the experiments which are Pareto optimal.

App labels are combined with Web labels using three different approaches, Stochastic label aggregation method, max-rank method and inverse-propensity method, these are discussed in section 3.2. Hyper-parameter α_{eng} controls how much Web and App engagement labels are used while combining them using stochastic label aggregation approach. We have experimented with $\alpha_{eng} \in \{0.2, 0.5, 0.8\}$. As α_{eng} increases, the amount of labels used from Web increases and from App it decreases. The baseline experiment is done using $\alpha_{eng} = 1$, meaning use 100% labels from Web. Max-Rank approach outperforms inverse propensity models and seems to be the best approach to combine App and Web datasets.

In subsequent experiments, we used $max - rank$ approach to combine Web and App labels, where we aim to add Query Product annotation and Query annotation labels.

Engagement labels are combined with Query Product dataset using stochastic label aggregation approach with different alphas (hyper-parameter). α_{rel} is used while combining. α_{rel} controls amount of dataset to use from each label source.

When using Engagement and Query Product annotation dataset all the experiments are Pareto optimal, but when we compare QIA NDCG experiment with $\alpha_{rel} = 0.2$ gives the highest lift. We used this setting for further experiments with Query annotation dataset. Parameter α_q is used while combining Query annotation data with Engagement and Query Product annotation combined dataset using stochastic Label aggregation approach. The experiments lying on the Pareto Frontier are highlighted in Bold in the table 2.

Method Name	Web NDCG	App NDCG	QIA NDCG	QA NDCG
Baseline $\alpha_{eng} = 1$	1	1	1	1
$\alpha_{eng} = 0.2$	**+0.73%**	**+2.1%**	-	-
$\alpha_{eng} = 0.5$	+0.82%	+2.04%	-	-
$\alpha_{eng} = 0.8$	+0.89%	+1.9%	-	-
max_rank	**+0.97%**	**+2.07%**	-	-
Inverse propensity	+0.88%	+2.01%	-	-
max_rank, $\alpha_{rel} = 0.2$	**+0.72%**	**+1.76%**	**+0.80%**	-
max_rank, $\alpha_{rel} = 0.5$	**+0.83%**	**+1.90%**	**+0.72%**	-
max_rank, $\alpha_{rel} = 0.8$	**+0.90%**	**+1.98%**	**+0.66%**	-
max_rank, $\alpha_{rel} = 0.2$, $\alpha_q = 0.2$	**+0.77%**	**+1.79%**	**+0.78%**	**+0.26%**
max_rank, $\alpha_{rel} = 0.2$, $\alpha_q = 0.5$	**+0.78%**	**+1.8%**	**+0.80%**	**+0.22%**

Table 2: Experimental Results

5 CONCLUSION

In this work, we have combined multiple label sources, aiming for a single LeToR model that optimizes for different aspects that are important for a customer. We have discussed, ways to leverage multiple labelled datasets available in E-commerce. We highlight ideas to come up with models and point out the Pareto optimal solutions. From the experiment results we have shown that Max Rank approach to combine Web and App datasets performs better than stochastic label aggregation approach. In other two experiments we have shown that the model performs better with smaller alpha's as Query-Product annotation and Query annotation datasets are small as compared to Engagement dataset.

REFERENCES

[1] Omar Alonso and Stefano Mizzaro. 2009. Relevance Criteria for E-Commerce: A Crowdsourcing-Based Experimental Analysis. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Boston, MA, USA) *(SIGIR '09)*. Association for Computing Machinery, New York, NY, USA, 760–761. https://doi.org/10.1145/1571941.1572115

[2] Michael E Barrett and Alan Levin. 2009. Enhanced popularity ranking. US Patent 7,565,367.

[3] David Carmel, Elad Haramaty, Arnon Lazerson, Liane Lewin-Eytan, and Yoelle Maarek. 2020. Why Do People Buy Seemingly Irrelevant Items in Voice Product Search? On the Relation between Product Relevance and Customer Satisfaction in ECommerce. In *Proceedings of the 13th International Conference on Web Search and Data Mining* (Houston, TX, USA) *(WSDM '20)*. Association for Computing Machinery, New York, NY, USA, 79–87. https://doi.org/10.1145/3336191.3371780

[4] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 785–794. https://doi.org/10.1145/2939672.2939785

[5] Arnon Lazerson David Carmel, Elad Haramaty and Liane Lewin-Eytan. 2020. Multi-Objective Ranking Optimization for Product Search Using Stochastic Label Aggregation. In *In Proceedings of The Web Conference 2020 (WWW'20), April 20–24, 2020, Taipei*. ACM, New York, NY, USA.

[6] Kalervo Järvelin and Jaana Kekäläinen. 2017. IR evaluation methods for retrieving highly relevant documents. In *ACM SIGIR Forum*, Vol. 51. ACM New York, NY, USA, 243–250.

[7] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased Learning-to-Rank with Biased Feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (Cambridge, United Kingdom) *(WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 781–789. https://doi.org/10.1145/3018661.3018699

[8] Shubhra Kanti Karmaker Santu, Parikshit Sondhi, and ChengXiang Zhai. 2017. On Application of Learning to Rank for E-Commerce Search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Shinjuku, Tokyo, Japan) *(SIGIR '17)*. Association for Computing Machinery, New York, NY, USA, 475–484. https://doi.org/10.1145/3077136.3080838

[9] Rohan Kumar, Mohit Kumar, Neil Shah, and Christos Faloutsos. 2018. Did We Get It Right? Predicting Query Performance in E-commerce Search. arXiv:1808.00239 [cs.IR]

[10] Tie-Yan Liu. 2009. Learning to Rank for Information Retrieval. In *Learning to Rank for Information Retrieval*, Vol. 3. Foundations and Trends® in Information Retrieval, 225–331. http://dx.doi.org/10.1561/1500000016

[11] Bo Long, Jiang Bian, Anlei Dong, and Yi Chang. 2012. Enhancing Product Search by Best-Selling Prediction in e-Commerce. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management* (Maui, Hawaii, USA) *(CIKM '12)*. Association for Computing Machinery, New York, NY, USA, 2479–2482. https://doi.org/10.1145/2396761.2398671

[12] R Timothy Marler and Jasbir S Arora. 2004. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization* 26, 6 (2004), 369–395.

[13] Michinari Momma, Alireza Bagheri Garakani, and Yi Sun. 2019. Multi-objective Relevance Ranking.. In *eCOM@ SIGIR*.

[14] Phong Nguyen, John Dines, and Jan Krasnodebski. 2017. A Multi-Objective Learning to re-Rank Approach to Optimize Online Marketplaces for Multiple Stakeholders.

[15] Matthew J Salganik and Duncan J Watts. 2008. Leading the herd astray: An experimental study of self-fulfilling prophecies in an artificial cultural market. *Social psychology quarterly* 71, 4 (2008), 338–355.

[16] Ning Su, Jiyin He, Yiqun Liu, Min Zhang, and Shaoping Ma. 2018. User Intent, Behaviour, and Perceived Satisfaction in Product Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (Marina Del Rey, CA, USA) *(WSDM '18)*. Association for Computing Machinery, New York, NY, USA, 547–555. https://doi.org/10.1145/3159652.3159714

[17] Vijay K Vemuri. 2015. Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction by Guido W. Imbens and Donald B. Rubin: New York, NY: Cambridge University Press, 2015, ISBN 978-0521885881, 644 pages, 60(hard-cover), 48 (eBook), 33.49(Kindleedition).

[18] Liang Wu, Diane Hu, Liangjie Hong, and Huan Liu. 2018. Turning Clicks into Purchases: Revenue Optimization for Product Search in E-Commerce. In *The 41st International ACM SIGIR Conference on Research amp; Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 365–374. https://doi.org/10.1145/3209978.3209993

[19] KALERVO J ARVELIN and JAANA KEK AL AINEN. 2002. Cumulated Gain-Based Evaluationof IR Techniques. *ACM Transactions on Information Systems,* 20, 4 (Oct. 2002), 422–446.