AnnoRank: A Comprehensive Web-Based Framework for Collecting Annotations and Assessing Rankings

Clara Rus
University of Amsterdam
Amsterdam, The Netherlands
c.a.rus@uva.nl

Gabrielle Poerwawinata
University of Amsterdam
Amsterdam, The Netherlands
g.poerwawinata@uva.nl

Andrew Yates
University of Amsterdam
Amsterdam, The Netherlands
a.c.yates@uva.nl

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
m.derijke@uva.nl

Abstract

We present AnnoRank, a web-based user interface (UI) framework designed to facilitate collecting crowdsource annotations in the context of information retrieval. AnnoRank enables the collection of explicit and implicit annotations for a specified query and a single or multiple documents, allowing for the observation of user-selected items and the assignment of relevance judgments. Furthermore, AnnoRank allows for ranking comparisons, allowing for the visualization and evaluation of a ranked list generated by different fairness interventions, along with its utility and fairness metrics. Fairness interventions in the annotation pipeline are necessary to prevent the propagation of bias when a user selects the top-k items in a ranked list. With the widespread use of ranking systems, the application supports multimodality through text and image document formats. We also support the assessment of agreement between annotators to ensure the quality of the annotations. AnnoRank is integrated with the Ranklib library, offering a vast range of ranking models that can be applied to the data and displayed in the UI. AnnoRank is designed to be flexible, configurable, and easy to deploy to meet diverse annotation needs in information retrieval. AnnoRank is publicly available as open-source software, together with detailed documentation, at https://github.com/ClaraRus/AnnoRank.

CCS Concepts

• **Information systems → Relevance assessment**; • **Human-centered computing → Visualization toolkits**.

Keywords

Annotation, Ranking, Fairness, User interface

ACM Reference Format:
Clara Rus, Gabrielle Poerwawinata, Andrew Yates, and Maarten de Rijke. 2024. AnnoRank: A Comprehensive Web-Based Framework for Collecting Annotations and Assessing Rankings. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3627673.3679174

1 Introduction

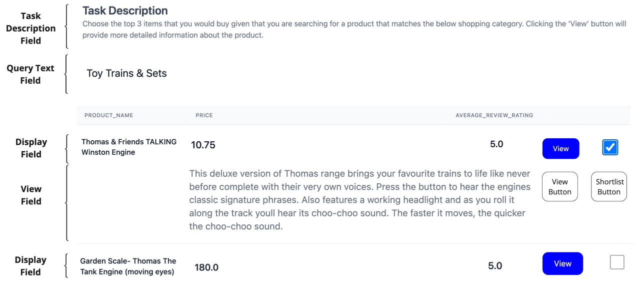
Search systems have become an integral part of everyday life, used for finding relevant items given various domains such as shopping items, images, and even candidates for a vacancy. Given their significance and broad applicability, assessing the responses of search systems is important.

In order to create or assess a search system, we need to collect relevance judgments. Relevance judgments can be collected through implicit or explicit feedback [7]. Explicit user feedback means that users explicitly assess the relevance of a document for a given query [8] using some predefined labels (e.g., on a scale of 1 to 5). Implicit user feedback is collected through the interactions performed while browsing the search results. For example, such interactions may include clicks, mouse movements, time spent to view an item's description or a person's profile.

There are several annotation tools that support the annotation of query-document pairs given a set of labels. Relevation! [10] is a tool that enables the annotation of query-document pairs given a set of configurable labels. FiRA [6] introduces finer-grained relevance grading at the level of passages and words. Similarly, DocTAG [4] and Doccano [15] offer the possibility of annotating documents at the passage level, as well as images. Neves and Ševa [16] provide an overview of text and document annotation tools. For example, WebAnno [22] is a comprehensive linguistic annotation tool that supports several functionalities, including POS tagging, semantic relations, and more. DocTAG [4] and Doccano [15] can be used as well in the annotation of text for NLP tasks.

While the landscape of annotation tools is rich, current options have limitations, often lacking the ability to gather implicit feedback and being confined to displaying only query-item pairs rather than a ranked list of items. However, understanding the user's behavior within a search platform is crucial, as it involves aspects such as search quality, relevance evaluation, user satisfaction, preference for their search, and interface design. Existing logging tools like Yasbil [1], LogUI [12], and Big Brother [17] can bridge this gap by providing the necessary functionalities to collect implicit feedback and generate logs based on user activities in a script that can be embedded in a web-based user interface. Therefore, in addition to the functionalities of existing annotation tools, AnnoRank uses

CIKM '24, October 21–25, 2024, Boise, ID, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679174



Task Description

Choose the top 3 items that you would buy given that you are searching for a product that matches the below shopping category. Clicking the 'View' button will provide more detailed information about the product.

Toy Trains & Sets

PRODUCT_NAME	PRICE	AVERAGE_REVIEW_RATING		
Thomas & Friends TALKING Winston Engine	10.75	5.0	View	☑
	This deluxe version of Thomas range brings your favourite trains to life like never before complete with their very own voices. Press the button to hear the engines classic signature phrases. Also features a working headlight and as you roll it along the track youll hear its choo-choo sound. The faster it moves, the quicker the choo-choo sound.			
Garden Scale- Thomas The Tank Engine (moving eyes)	180.0	5.0	View	□

Figure 1: Interaction Annotation UI adapted for the Amazon dataset. The View Button is used to make the View Field of the corresponding item visible, while collecting the clicks. The Shortlist Button is used to select the top-k relevant items.

existing logging tools (e.g., Big Brother) to support the collection of implicit feedback, especially in the case of the collection of top-k items for a single query by recording annotators' item selections.

Annotation tasks can be performed using crowd-sourcing platforms such as Prolific,[1] MTurk,[2] and more. Other than providing a tool for performing relevance judgment tasks, AnnoRank is also integrated with ranking and fairness libraries, which makes it more robust when such use-cases are needed. It also supports visualisation and comparisons of rankings together with computed metrics and inter-annotator agreements.

In summary, AnnoRank offers three main UI functionalities for collecting annotations: (i) displaying a query-ranked result list pair, (ii) displaying a query-item pair, and (iii) comparing two ranked lists given a query.

Researchers or users with minimal web development experience can utilize the tool, as the UI can easily be configured using specific configuration files in JSON format for each functionality in AnnoRank. The interface is also designed to be user-friendly and intuitive, allowing crowdsource annotators with varying expertise levels to participate in assessments. Integration with external libraries, such as Ranklib, fairness libraries, and interrater reliability measurement tools, enables support for critically assessing annotations and thereby enabling fairness interventions in the ranking system pipeline and mitigating biases.

2 Tool Description

AnnoRank, a web application tool, serves two primary functions. The first function allows for collecting explicit and implicit annotations for a ranked list of documents and pairs of queries and documents. The annotated feedback output can subsequently be used to predict relevant feedback rankings for a given query, determine user interest based on their previous actions, and prefetch relevant documents for similar queries. Given the potential variability in ranking results for a single query, AnnoRank allows researchers to view the collected annotations, compare two rankings, and examine the corresponding evaluation metrics. By providing adequate ranking comparison, the tool may capture users' preferences toward the ranking model. The process for annotating tasks can be described as follows: users are first prompted to provide a user ID, as user feedback is specific to a user. Following this, the user interface (UI) will present the applicable annotation assignment based on the chosen functionality. Upon completion of the annotation task, annotators can choose to participate in an exit survey where annotators can provide their opinions. Collecting annotators' opinions on completing a task can help highlight the potential challenges while performing the tasks and increase engagement between annotators and researchers.

2.1 Interaction Annotation UI

In comparison to existing annotation tools, AnnoRank has the option to display a query-ranked relevant documents pair, for which one can configure the UI to collect both implicit and explicit annotations. Figure 1 shows how the UI will display the query-ranked relevant documents pair, with the task description and the query at the top of the page followed by the ranked list of items. Depending on the type of feedback that one desires to collect, the UI can be configured to display a *check-box* with which the user can select the top-k relevant items given the query and a *View* button with which the users can view more information about the item. When clicking on the *View* button, the field of the item will expand as shown in Figure 1 *View Field*. This design aims to simulate a scenario where users browse the ranked list of items, clicking on items to view more information and make an informed decision. This tool uses timestamps to measure and record implicit feedback, such as time spent on each ranking item and the order of the selected top-k items. The web activities by the user can also be inspected thoroughly with the help of the Big Brother [17] logging tool. Depending on the Big Brother configuration in AnnoRank, the tool will generate a log file of mouse activities, such as clicking on a specific HTML item and mouse movement on the web page.

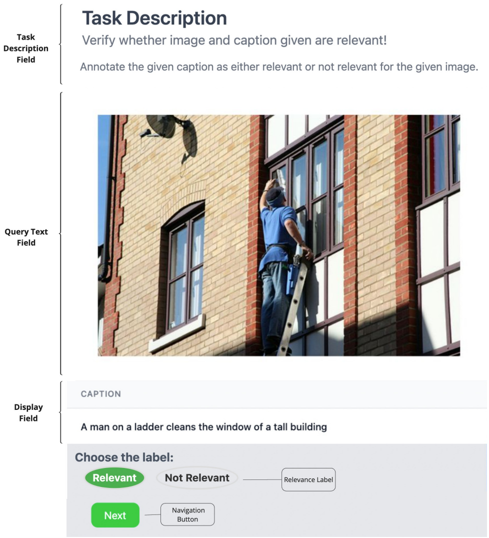
2.2 Score Annotation UI

Similar to existing annotation tools, AnnoRank supports the collection of annotations for a query-item pair. Figure 2 shows how the UI will display the query-item pair, with the query at the top of the page followed by the item. Given a query, the user can explicitly choose a label using the label buttons at the bottom of the page to indicate the item's relevance. These labels can be customized to cover any value range or boolean values, with numerical and textual label types based on the assignment's requirements. The mouse activities in this part of the UI can also be monitored with the Big Brother logging tool.

2.3 Ranking Comparison UI

Similar to [9, 18], AnnoRank has the option to compare two rankings side by side. Figure 3 shows the comparison UI, which displays in the top of the page the query, and below the two rankings to be compared. This comparative view is instrumental for researchers, providing a means to critically evaluate and contrast the output of various ranking algorithms or examine the effects of applied fairness measures. By comparing ranking models, it can also help researchers determine the complexity of user preferences, assess different approaches in generating recommendations based on user-specific features.

[1] https://www.prolific.com/
[2] https://www.mturk.com/



Task Description

Verify whether image and caption given are relevant!

Annotate the given caption as either relevant or not relevant for the given image.

CAPTION

A man on a ladder cleans the window of a tall building

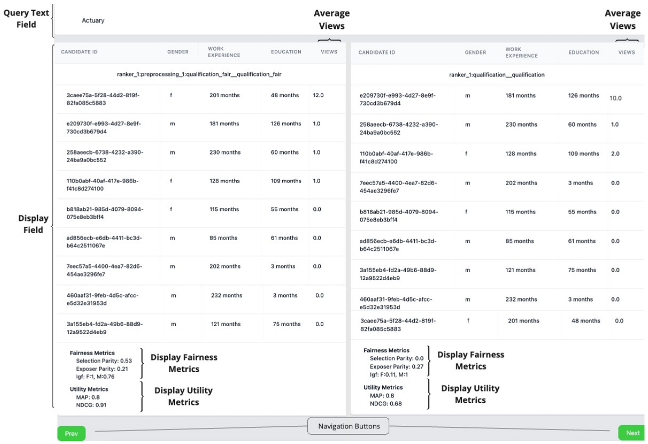
Choose the label:

Figure 2: Score Annotation UI for annotating images.

Additionally, the UI displays the average views collected for each item, considering the displayed ranking. The tool is integrated with the Pytrec Eval [19] library, which supports all utility metrics defined by the library. The fairness metrics that the tool supports are selection parity, parity of exposure, and IGF (in group fairness) [20]. Lower values mean the ranking offers equal representation/exposure among the groups, while for IGF higher values mean more in group fairness. It is to be noted that the metrics are not meant to be shown to the annotator.

2.4 Additional Support

Users tend to prefer the first top items without scrolling to the bottom of the page [5]. Top positions receive more attention from the user, introducing bias in the ranking system. For this reason, AnnoRank offers additional support for applying fairness interventions in the pipeline of a ranking system. The tool’s fairness interventions ready to use are FA*IR [23], a post-processing fairness intervention, and CIF-Rank [21], a pre-processing fairness intervention. This is especially useful for researching fairness in ranking. In addition, the tool offers support for training a ranking model and applying the model to the data to be displayed. The ready to use ranking models (e.g., RankNet [2], ListNet [3]) make use of the implementation provided by the Ranklib[3] library. AnnoRank supports applying the fairness interventions on the input/output of the ranker, as well as applying a combination of fairness interventions. More details about using the rankers and the fairness interventions can be found in the provided documentation, along with easy-to-follow steps about integrating other ranking toolkits or fairness interventions. To ensure the quality of the inter-annotator reliability toward the assigned tasks, we use the Python agreement[4] library to compute the Kappa statistics such as Krippendorff’s, Cohen’s, and Weighted Cohen’s kappas. These kappas’ values range from −1 to +1, with +1 indicating perfect reliability, while −1 indicating systematic disagreements among raters [13].



Actuary

CANDIDATE ID	GENDER	WORK EXPERIENCE	EDUCATION	VIEWS
ranker_1:preprocessing_1:qualification_fair__qualification_fair				
3caee75a-5f28-44d2-819f-82fa085c5883	f	201 months	48 months	12.0
e209730f-e993-4d27-8e9f-730cd3b679d4	m	181 months	126 months	1.0
258aeecb-6738-4232-a390-24ba9a0bc552	m	230 months	60 months	1.0
110b0abf-40af-417e-986b-f41c8d274100	f	128 months	109 months	1.0
b818ab21-985d-4079-8094-075e8eb3bff4	f	115 months	55 months	0.0
ad856ecb-e6db-4411-bc3d-b64c2511067e	m	85 months	61 months	0.0
7eec57a5-4400-4ea7-82d6-454ae3296fe7	m	202 months	3 months	0.0
460aaf31-9feb-4d5c-afcc-e5d32e31953d	m	232 months	3 months	0.0
3a155eb4-fd2a-49b6-88d9-12a9522d4eb9	m	121 months	75 months	0.0

Fairness Metrics
Selection Parity: 0.53
Exposer Parity: 0.21
Igf: F:1, M:0.76

Utility Metrics
MAP: 0.8
NDCG: 0.91

CANDIDATE ID	GENDER	WORK EXPERIENCE	EDUCATION	VIEWS
ranker_1:qualification__qualification				
e209730f-e993-4d27-8e9f-730cd3b679d4	m	181 months	126 months	10.0
258aeecb-6738-4232-a390-24ba9a0bc552	m	230 months	60 months	1.0
110b0abf-40af-417e-986b-f41c8d274100	f	128 months	109 months	2.0
7eec57a5-4400-4ea7-82d6-454ae3296fe7	m	202 months	3 months	0.0
b818ab21-985d-4079-8094-075e8eb3bff4	f	115 months	55 months	0.0
ad856ecb-e6db-4411-bc3d-b64c2511067e	m	85 months	61 months	0.0
3a155eb4-fd2a-49b6-88d9-12a9522d4eb9	m	121 months	75 months	0.0
460aaf31-9feb-4d5c-afcc-e5d32e31953d	m	232 months	3 months	0.0
3caee75a-5f28-44d2-819f-82fa085c5883	f	201 months	48 months	0.0

Fairness Metrics
Selection Parity: 0.0
Exposer Parity: 0.27
Igf: F:0.11, M:1

Utility Metrics
MAP: 0.8
NDCG: 0.68

Figure 3: Ranking Comparison UI. On the bottom of each ranking, the Evaluation Metrics are displayed.

3 Implementation and Usage

3.1 Requirements

We use docker containers to handle our applications and database. Docker[5] improves reproducibility in software and web engineering research [14] and also simplifies the deployment of our web applications. With docker, AnnoRank can be deployed in most hosting machines. AnnoRank is built with the Python Flask[6] framework and employs MongoDB[7] as the underlying database.

3.2 Usage and Workflow

In order to launch the app, it is necessary to define configuration files that specify the dataset and other variables required for the UI components. Afterward, by calling the *run_apps.sh* script, it will trigger the creation of the docker containers. A new database is automatically generated with the necessary collections for each dataset. Once the app is up and running, the web apps can be accessed through the following links:

- **Interaction Annotation UI:** http://localhost:5000/start_ranking/<exp_id>
- **Score Annotation UI:** http://localhost:5003/start_annotate/<exp_id>
- **Ranking Comparison Visualise UI:** http://localhost:5001/start_compare/<exp_id>,

The ports are defined in the *docker-compose* file; if needed, one can change the ports to other values. The *exp_id* is the ID of the assignment to be displayed to the users. An assignment can consist of one or more assessments, that one should define in an experiment configuration file. Each UI app has a specific format for defining the assessments. Optionally, one can adapt the UI to include extra

[3] https://sourceforge.net/p/lemur/wiki/RankLib/
[4] https://pypi.org/project/agreement
[5] https://www.docker.com/
[6] https://flask.palletsprojects.com/en/3.0.x/
[7] https://www.mongodb.com/

requirements in the task description for each assessment. The assessments are displayed in a random order to each user, and not in the order defined in the configuration file.

AnnoRank supports adding attention-check tasks to ensure that the collected annotations are of sufficient quality, and the crowd-source annotators are actively engaged in the task and not simply rushing through it. Researchers may also filter out responses from annotators who do not perform as expected.

If multiple experiments needed to be run on the same dataset, one needs to define a configuration file for each experiment and only change the *exp_id* in the link. To run multiple experiments on different datasets, one needs to run a docker instance for each dataset and define the experiment to run in a configuration file for each dataset. The collected data can be exported as either JSON or CSV files. Researchers can execute the *iaa_metrics.py* app at any time to generate the computation of the kappa's statistics.

3.3 Flexibility

AnnoRank is designed to be flexible, configurable, and easy to use. Adapting the UI of the app to specific requirements does not require prior knowledge of programming as one only needs to define the UI varied components and functionalities in the configuration file. Still, there are some specific components that are dependent on the UI functionality. For example, for the Interaction Annotation UI one can opt to include buttons and specify the information to be displayed under the expanded tab. For the Score Annotation UI, one can define the score range of the score bar. Finally, for the comparison functionality, one can define which metrics to display.

Adapting the tool to work with a new dataset is straightforward as it only involves defining a Python *data_reader* class that should convert the dataset format to a tabular format. AnnoRank can be used with any data format. Next, AnnoRank supports the automatic insertion of the data into the database as long as the mandatory fields are present in the tabular format of the dataset. On top of the mandatory required fields, the tool automatically adds the rest of the columns present in the data as dynamic fields. The documentation[8] provides ready to use examples of integrating a new dataset into the AnnoRank tool as well as a step by step tutorial (Section 7 & 8).

As pointed out above, AnnoRank is integrated with the Ranklib library and several ready-to-use fairness interventions. This integration allows users to employ advanced ranking algorithms while ensuring that search results are relevant and fair. Incorporating the user interface ranking tool with Ranklib or other ranking libraries may widen the possibility of using more ranking models. Also, the fairness intervention library helps prevent biases in the results, promoting inclusivity. Integrating alternative retrieval toolkits into AnnoRank is straightforward, ensuring flexibility and ease of use by only writing an additional wrapper class. This modular approach facilitates customization and adaptation to various researcher needs.

AnnoRank can easily be adapted to various annotation scenarios given that the query placeholder and item(s) placeholder can display both text and image. For example, one could use AnnoRank to judge the generated responses of a RAG system as well as the retrieved list.

3.4 Usability Study

So far, AnnoRank was used to collect explicit and implicit feedback for a recruitment dataset as part of the FINDHR[9] initiative of understanding the effect of ranking strategies in recruitment. To further test AnnoRank in real environments, we conducted a small-scale usability study. Among other things, the participants were asked to follow the assignment presented in Figure 1 and 2. After completion of the assignment we conducted an interview to evaluate their experience. In total, we had 14 participants with a computer science, management, and physics educational background, out of which 3 tested only the set-up. Each assignment was tested by 5 participants.

Following [11], we conducted interviews with the participants in which we asked them to evaluate the assignment in terms of appearance, content, navigation, and functionality. 9 out of 11 participants reported that AnnoRank has an intuitive, straightforward and usable user interface. Regarding the annotation assignments, participants reported that the task was easy to follow and straightforward to complete. Additionally, we evaluated the technical setup of AnnoRank by asking participants to download AnnoRank and start the app. The participants considered the setup straightforward and easy to follow. The install time was considered to be reasonable.

In response to the feedback received during the usability study, we performed the following improvements: made the *View* Button more visible and added the possibility to automatically highlight terms from the query in the item's text to ease the annotation task. Installing the app on Windows required extra requirements that we added in the ReadMe after conducting the usability study and installing docker on Mac OS 11.5 failed.

4 Conclusion

To address the lack of annotation tools for a query-ranked list of items pair, we have proposed a web-based user interface (UI) framework that supports the collection of both explicit and implicit annotations in this setting, together with integration with various ranking, fairness intervention, and interrater agreement library. AnnoRank is designed to be flexible, configurable, and easy to use to meet diverse requirements and a larger audience.

The tool offers three main UI functionalities: (i) the option to collect both explicit and implicit annotation for a ranked list of items given the displayed query, (ii) the option to collect graded relevance annotations for an item given a query and (iii) the option to compare two rankings. Finally, we conducted a usability study to test AnnoRank in real environments, which concluded that AnnoRank is easy to install and intuitive to use. The open-source software together with detailed documentation are publicly available.

As for limitations, the current implementation of the supported fairness metrics can be computed only for binary groups; thus, in the future, the metrics should be adapted for more than two values. AnnoRank offers the possibility to collect the top-k relevant items while also collecting the order in which the user selected an item, from which one can infer that the first selected item is more relevant than the second. In the future, we intend to adapt AnnoRank to support graded relevance annotations for the query-ranked result list pair UI functionality.

[8] https://github.com/ClaraRus/AnnoRank/blob/main/external-resources/Anno_Rank_Documentation.pdf

[9] https://findhr.eu/

Acknowledgments

We thank our reviewers for valuable feedback.

This research was supported by the FINDHR (Fairness and Intersectional Non-Discrimination in Human Recommendation) project that received funding from the European Union's Horizon Europe research and innovation program under grant agreement No 101070212, and by he Dutch Research Council (NWO), under project numbers VI.Vidi.223.166, 024.004.022, NWA.1389.20.183, and KICH3.LTP.20.006.

All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

References

[1] Nilavra Bhattacharya and Jacek Gwizdka. 2021. YASBIL: Yet Another Search Behaviour (and) Interaction Logger. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2585–2589. https://doi.org/10.1145/3404835.3462800

[2] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to Rank Using Gradient Descent. In *Proceedings of the 22nd International Conference on Machine Learning*. 89–96. https://doi.org/10.1145/1102351.1102363

[3] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to Rank: From Pairwise Approach to Listwise Approach. In *Proceedings of the 24th International Conference on Machine Learning*. 129–136. https://doi.org/10.1145/1273496.1273513

[4] Fabio Giachelle, Ornella Irrera, and Gianmaria Silvello. 2022. DocTAG: A Customizable Annotation Tool for Ground Truth Creation. In *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II* (Stavanger, Norway). Springer-Verlag, Berlin, Heidelberg, 288–293. https://doi.org/10.1007/978-3-030-99739-7_35

[5] Laura A. Granka, Thorsten Joachims, and Geri Gay. 2004. Eye-tracking Analysis of User Behavior in WWW Search. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 478–479. https://doi.org/10.1145/1008992.1009079

[6] Sebastian Hofstätter, Markus Zlabinger, Mete Sertkan, Michael Schröder, and Allan Hanbury. 2020. Fine-Grained Relevance Annotations for Multi-Task Document Ranking and Question Answering. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (Virtual Event, Ireland) *(CIKM '20)*. Association for Computing Machinery, New York, NY, USA, 3031–3038. https://doi.org/10.1145/3340531.3412878

[7] Jeff Huang, Ryen W. White, and Susan Dumais. 2011. No Clicks, No Problem: Using Cursor Movements to Understand and Improve Search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. Association for Computing Machinery, New York, NY, USA, 1225–1234. https://doi.org/10.1145/1978942.1979125

[8] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately Interpreting Clickthrough Data as Implicit Feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Salvador, Brazil) *(SIGIR '05)*. Association for Computing Machinery, New York, NY, USA, 154–161. https://doi.org/10.1145/1076034.1076063

[9] Kevin Martin Jose, Thong Nguyen, Sean MacAvaney, Jeffrey Dalton, and Andrew Yates. 2021. DiffIR: Exploring Differences in Ranking Models' Behavior. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 2595–2599. https://doi.org/10.1145/3404835.3462784

[10] Bevan Koopman and Guido Zuccon. 2014. Relevation! An Open Source System for Information Retrieval Relevance Assessment. In *Proceedings of the 37th international ACM SIGIR Conference on Research & Development in Information Retrieval*. 1243–1244. https://doi.org/10.1145/2600428.2611175

[11] Michelle Lobchuk, Lisa Hoplock, Nicole Harder, Marcia Friesen, Julie Rempel, and Prachotan Reddy Bathi. 2023. Usability Testing of a Web-Based Empathy Training Portal: Mixed Methods Study. *JMIR Formative Research* 7 (2023), e41222. https://doi.org/10.2196/41222

[12] David Maxwell and Claudia Hauff. 2021. LogUI: Contemporary Logging Infrastructure for Web-based Experiments. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43*. Springer, 525–530. https://doi.org/10.1007/978-3-030-72240-1_59

[13] Mary L. McHugh. 2012. Interrater Reliability: The Kappa Statistic. *Biochemia Medica* 22 (2012), 276–282. https://api.semanticscholar.org/CorpusID:5421278

[14] Dirk Merkel. 2014. Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux Journal* 2014, 239, Article 2 (mar 2014). https://www.linuxjournal.com/content/docker-lightweight-linux-containers-consistent-development-and-deployment

[15] Hiroki Nakayama, Takahiro Kubo, Junya Kamura, Yasufumi Taniguchi, and Xu Liang. 2018. doccano: Text Annotation Tool for Human. https://github.com/doccano/doccano Software available from https://github.com/doccano/doccano.

[16] Mariana Neves and Jurica Ševa. 2021. An Extensive Review of Tools for Manual Annotation of Documents. *Briefings in Bioinformatics* 22, 1 (2021), 146–163. https://doi.org/10.1093/bib/bbz130

[17] Harrisen Scells, Jimmy, and Guido Zuccon. 2021. Big Brother: A Drop-in Website Interaction Logging Service. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2590–2594. https://doi.org/10.1145/3404835.3462781

[18] Paul Thomas and David Hawking. 2006. Evaluation by Comparing Result Sets in Context. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management* (Arlington, Virginia, USA) *(CIKM '06)*. Association for Computing Machinery, New York, NY, USA, 94–101. https://doi.org/10.1145/1183614.1183632

[19] Christophe Van Gysel and Maarten de Rijke. 2018. Pytrec_eval: An Extremely Fast Python Interface to trec_eval. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 873–876. https://doi.org/10.1145/3209978.3210065

[20] Ke Yang, Vasilis Gkatzelis, and Julia Stoyanovich. 2019. Balanced Ranking with Diversity Constraints. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19)*. 6035–6042. https://www.ijcai.org/proceedings/2019/0836.pdf

[21] Ke Yang, Joshua R. Loftus, and Julia Stoyanovich. 2020. Causal Intersectionality for Fair Ranking. *arXiv preprint arXiv:2006.08688* (2020). https://arxiv.org/pdf/2006.08688

[22] Seid Muhie Yimam, Iryna Gurevych, Richard Eckart de Castilho, and Chris Biemann. 2013. Webanno: A Flexible, Web-based and Visually Supported System for Distributed Annotations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. ACL, 1–6. https://aclanthology.org/P13-4001

[23] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. 2017. FA*IR: A Fair Top-k Ranking Algorithm. In *Proceedings of the 2017 ACM Conference on Information and Knowledge Management*. ACM, 1569–1578. https://doi.org/10.1145/3132847.3132938