Vis-Trec: A System for the In-depth Analysis of `trec_eval` Results

Mahtab Tamannaee
Ryerson University, LS3
Toronto, Canada
mtamannaee@ryerson.ca

Negar Arabzadeh
Ryerson University, LS3
Toronto, Canada
narabzad@ryerson.ca

Ebrahim Bagheri
Ryerson University, LS3
Toronto, Canada
bagheri@ryerson.ca

ABSTRACT

We introduce Vis-Trec[1], an open-source cross-platform system, which provides the capability to perform in-depth analysis of the results obtained from trec-style evaluation campaigns. Vis-Trec allows researchers to dig deeper in their evaluations by providing various visualizations of the results based on performance percentiles, query difficulty, and comparative analysis of different methods using help-hurt diagrams at the query level. It also automatically organizes the obtained results in tabular LaTeX format that can be used for reporting evaluation findings. The added benefit for Vis-Trec is that it has been developed in Python and is extensible by other developers. The source code along with a functional version of the program are released to the public.

ACM Reference Format:
Mahtab Tamannaee, Negar Arabzadeh, and Ebrahim Bagheri. 2020. Vis-Trec: A System for the In-depth Analysis of `trec_eval` Results. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3397271.3401412

1 INTRODUCTION

The systematic analysis of the performance of an Information Retrieval (IR) method relies not only on macro-level metrics, but also on granular analysis of the results at finer-grained levels such as separation of the performance of the method across a range of soft to hard queries or even analyzing the performance of each individual query [5, 10]. The IR community has already standardized the process of evaluating the performance of methods that relate to various forms of ad hoc retrieval through the `trec_eval` tool[2]. This tool computes a host of metrics at both collective and query levels, given (1) a list of queries, (2) their set of relevant documents, known as *qrels*, and (3) a *run* file representing the performance of the system under test. Researchers often employ the outcomes reported by `trec_eval` to perform additional more in-depth analysis of performance. While there exist many useful toolkits [1, 3, 7–9], the process of performing the additional analysis is often either performed using in-house code from the researchers' labs or through a collection of scripts such as the minir-plots[3] that have been shared with the community as a result of a personal project.

Based on our experience in running IR evaluations, we found that a streamlined software that would wrap around the `trec_eval` tool and automatically perform additional experiments would be immensely helpful as it will (1) save time by eliminating the need to run various evaluation scripts many times, (2) provide various forms of analysis on the results through a visual interface; hence, minimizing possible errors that might happen when running multiple scripts, and (3) generate uniform and easy to understand visualizations that can be seamlessly used when reporting.

The objective of this paper is to offer such a tool to the community, called Vis-Trec[4], which contributes to streamlining the computation of three widely used analysis *tasks* in IR, namely (Task 1) *IR evaluation metrics visualization* enhancing the understanding of performance of a system at both individual query and collective levels and allowing for the visual comparison of performance of various baselines and the system under test; (Task 2) *Query analysis*, which analyzes which and to what extent various queries have been improved or declined compared to a chosen baseline; and (Task 3) *Query difficulty analysis*, which leads to distinguishing hard from soft queries, and providing insight into the performance of the system under different query difficulties. Task 3 cross-cuts the other two in the sense that by identifying hard queries, we can perform Tasks 1 and 2 on a range of hard and soft queries.

In the process of implementing Vis-Trec, we have focused on providing a unified visual software to afford the three mentioned tasks by dispensing two types of visual outputs including (1) *Images* of plots relating to different tasks; and (2) LaTeX *code* for tables consisting of the retrieval results and their in-depth comparison. Vis-Trec wraps around the `trec_eval` tool and provides a visual GUI representation offering several benefits such as; (1) convenient and user-friendly environment for running experiments over many iterations; (2) reducing the chances of unintended errors that may arise when multiple scripts are used and run files are copied across different locations; (3) clear organization of experimental findings by logging the in-depth analyses with timestamps; and, (4) easily extensible with new functionality as it is return with a clear structure in Python and is released as open-source.

2 SYSTEM OVERVIEW

In this section, we briefly describe the functionality provided by Vis-Trec with regards to the three tasks introduced in the previous section.

[1]Tutorial available on YouTube: https://github.com/usnistgov/trec_eval
[2]https://github.com/usnistgov/trec_eval

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGIR '20, July 25–30, 2020, Virtual Event, China
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401412

[3]https://github.com/laura-dietz/minir-plots
[4]Code available at https://github.com/mtamannaee/Vis-Trec

2.1 IR Evaluation Metrics Visualization

One of the most helpful ways of understanding the performance of a retrieval system is by visually representing its performance on various performance percentiles. Vis-Trec allows the user to easily define desirable percentiles and visualize a comparative analysis of the performance of one or more retrieval methods based on these percentiles. In Vis-Trec, researchers can conveniently evaluate their work across different percentiles by simply defining different percentile groups based on (1) a metric of interest and (2) a base retrieval method. Based on the defined percentiles, Vis-Trec will identify the queries for each percentile, matches the queries across different retrieval methods, and plots the performance of the various methods on a comparative basis. Figure 1 shows a plot generated by Vis-Trec where MAP has been chosen as the metric and four baselines are compared according to four user-defined percentiles.

2.2 Query Help-Hurt Analysis

In order to be able to understand where a retrieval method has been successful and where it has not, researchers need to understand retrieval performance at the query level. Vis-Trec provides the ability to visualize how a retrieval method performs compared to a selected baseline according to an evaluation metric of choice on a query by query basis. The user can select the retrieval method that is under test as well as a baseline method and also specify what metric would be used for comparing the two methods' performance and on this basis, Vis-Trec would automatically plot a diagram showing how queries are helped or hurt in the retrieval method that is under test. Figure 2 depicts a sample Vis-Trec diagram comparing the performance of two retrieval methods based on MAP. It should be noted that Vis-Trec computes the delta of the performance of the two methods based on the selected metric and reports it on the y-axis.

2.3 Query Difficulty Analysis

When evaluating a retrieval method, researchers need to understand how the method performs across a range of query types. Most methods have a set of queries for which they cannot perform well, known as difficult or hard queries [2, 4]. When working on improving retrieval methods, researchers would need to understand how their work is impacting the hard queries and whether they are being helped or not. To this end, Vis-Trec allows its users to input their definition of a hard query and automatically identifies the collection of queries that match this definition. For instance, users can specify that they would like hard queries to be poor performing queries whose MAP values are less than 5% according to the Sequential Dependence Model (SDM) [6]. The metric's cutoff value, the metric used, and the baseline method to be used for defining hard queries can all be determined by the user. Vis-Trec would then allow the user to only focus on this subset of queries.

Following the example of user defining the hard queries as those whose SDM[6] MAP values are less than 0.05, the Vis-Trec can further provide the user with percentile-based IR evaluation or help-hurt analysis on hard queries by comparing the performance of the retrieval methods on a metric of evaluation.

2.4 Vis-Trec Artifacts

Vis-Trec provides two methods for accessing the outcomes of the above tasks: (1) Images, which are saved on a directory of user's choice with appropriate timestamps to avoid overwriting or losing previous runs. These images are saved in PNG format, e.g., diagrams in Figures 1 and 2 were generated this way, and (2) LaTeX code for reporting results in table format. Figure 3 shows LaTeX code that is automatically generated by Vis-Trec. Generating LaTeX code by Vis-Trec save researchers from the hassle of creating tables, transferring results into the appropriate format and decreases the chances of incorrectly reporting results.

2.5 Graphical User Interface

Vis-Trec provides a graphical user interface for performing two steps: Step 1: *Trec Evaluation*, which runs `trec_eval` on the provided run files and collects the results, and Step 2: *Analysis and Visualization*, which allows the user to perform the tasks mentioned in the previous sections. Figure 4 provides a snapshot of Vis-Trec's user interface. The user has the option of selecting a folder with various run files as well as identifying the appropriate relevance judgement file in the top box of the GUI. The user can subsequently run `trec_eval` directly from Vis-Trec. Once `trec_eval` results are obtained, the two tabs related to *All Queries* and *Hard Queries* will be activated where the user can perform various analysis tasks as explained earlier. Figure 4 shows the tasks that can be performed over all queries. In addition, when the Hard Queries tab is chosen, users have the option to define hard queries and then perform additional analysis on these selected queries.

3 CONCLUDING REMARKS

This demo paper introduces the Vis-Trec tool whose objective is to allow researchers to seamlessly analyze the outcomes of the `trec_eval` tool with the comfort of a graphical user interface with the option to easily define query percentiles and query difficulty ranges. Vis-Trec produces graphical plots of retrieval methods based on different metrics and query groups. It is also able to automatically generate LaTeX code that can be used for reporting purposes. The tool is released as open source and is developed in Python and hence can easily be extended with additional functionality.

Currently, we are extending our work in two complementary directions:

- We are working on a web-based system that would allow researchers to run their evaluation campaigns online and generate diagrams and prepare results. This would be convenient as it will allow for collaborative work on experiments and discussion of findings.
- As a part of the above, we are planning to create a public web-based platform to allow researchers to openly share their run files with the community. This will immensely help with reproducibility of results and comparison against baseline methods reported in the literature.

REFERENCES

[1] Enrique Amigó, Jorge Carrillo-de Albornoz, Mario Almagro-Cádiz, Julio Gonzalo, Javier Rodríguez-Vidal, and Felisa Verdejo. 2017. Evall: Open access evaluation for

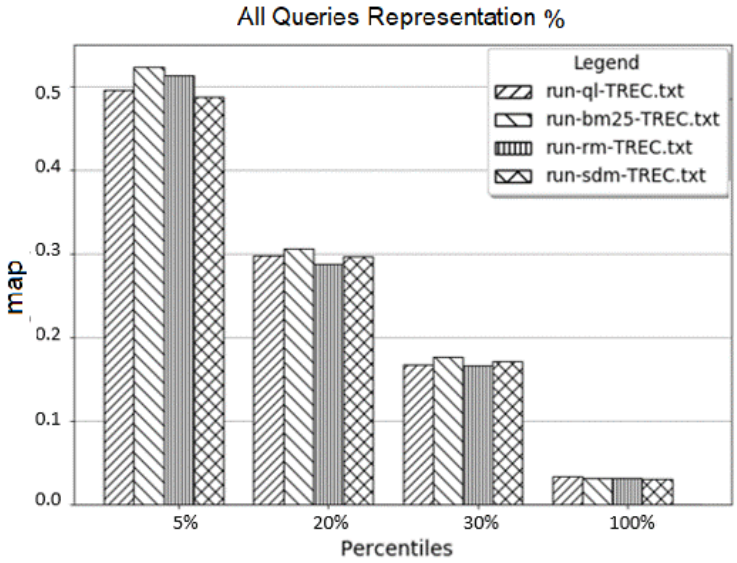

Figure. 1 Vis-Trec Diagrams: all queries percentile-based visualization.

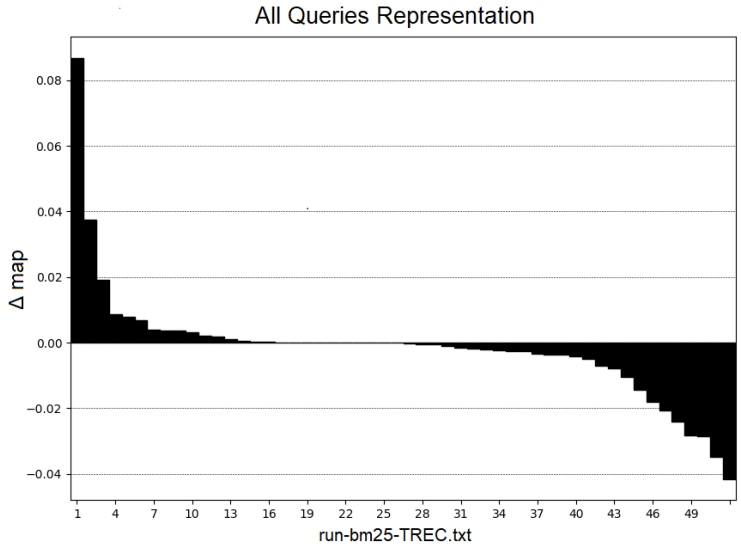

Figure. 2 Vis-Trec Diagrams: all queries help-hurt diagram.

information access systems. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1301–1304.

[2] Negar Arabzadeh, Fattane Zarrinkalam, Jelena Jovanovic, and Ebrahim Bagheri. 2020. Neural Embedding-based Metrics for Pre-Retrieval Query Performance Prediction. In *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2019, Proceedings,*.

[3] Leif Azzopardi, Paul Thomas, and Alistair Moffat. 2019. cwl_eval: An evaluation tool for information retrieval. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1321–1324.

[4] David Carmel and Elad Yom-Tov. 2010. Estimating the query difficulty for information retrieval. *Synthesis Lectures on Information Concepts, Retrieval, and Services* 2, 1 (2010), 1–89.

[5] Kalervo Järvelin and Jaana Kekäläinen. 2000. IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 41–48.

[6] Donald Metzler and W Bruce Croft. 2005. A Markov random field model for term dependencies. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 472–479.

[7] Joao Palotti, Harrisen Scells, and Guido Zuccon. 2019. Trectools: an open-source python library for information retrieval practitioners involved in trec-like campaigns. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1325–1328.

[8] Kevin Roitero, Eddy Maddalena, Yannick Ponte, and Stefano Mizzaro. 2018. IRevalOO: An Object Oriented Framework for Retrieval Evaluation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 913–916.

```
\begin{tabular}{lr}
\hline
                    &   All Queries (map) \\
\hline
 run-ql-TREC.txt    &             0.09975  \\
 run-bm25-TREC.txt  &             0.101338 \\
 run-rm-TREC.txt    &             0.0975404 \\
 run-sdm-TREC.txt   &             0.0973654 \\
\hline
\end{tabular}
```

	All Queries (map)
run-ql-TREC.txt	0.09975
run-bm25-TREC.txt	0.101338
run-rm-TREC.txt	0.0975404
run-sdm-TREC.txt	0.0973654

```
\begin{tabular}{lr}
\hline
                    &   Hard Queries (map) \\
\hline
 run-ql.adhoc       &             0.0212061 \\
 run-bm25.adhoc     &             0.0187636 \\
 run-rm.adhoc       &             0.0206879 \\
 run-sdm.adhoc      &             0.0190364 \\
\hline
\end{tabular}
```

	Hard Queries (map)
run-ql.adhoc	0.0212061
run-bm25.adhoc	0.0187636
run-rm.adhoc	0.0206879
run-sdm.adhoc	0.0190364

Figure. 3 Two sample LaTeX codes and corresponding tables generated by Vis-Trec.

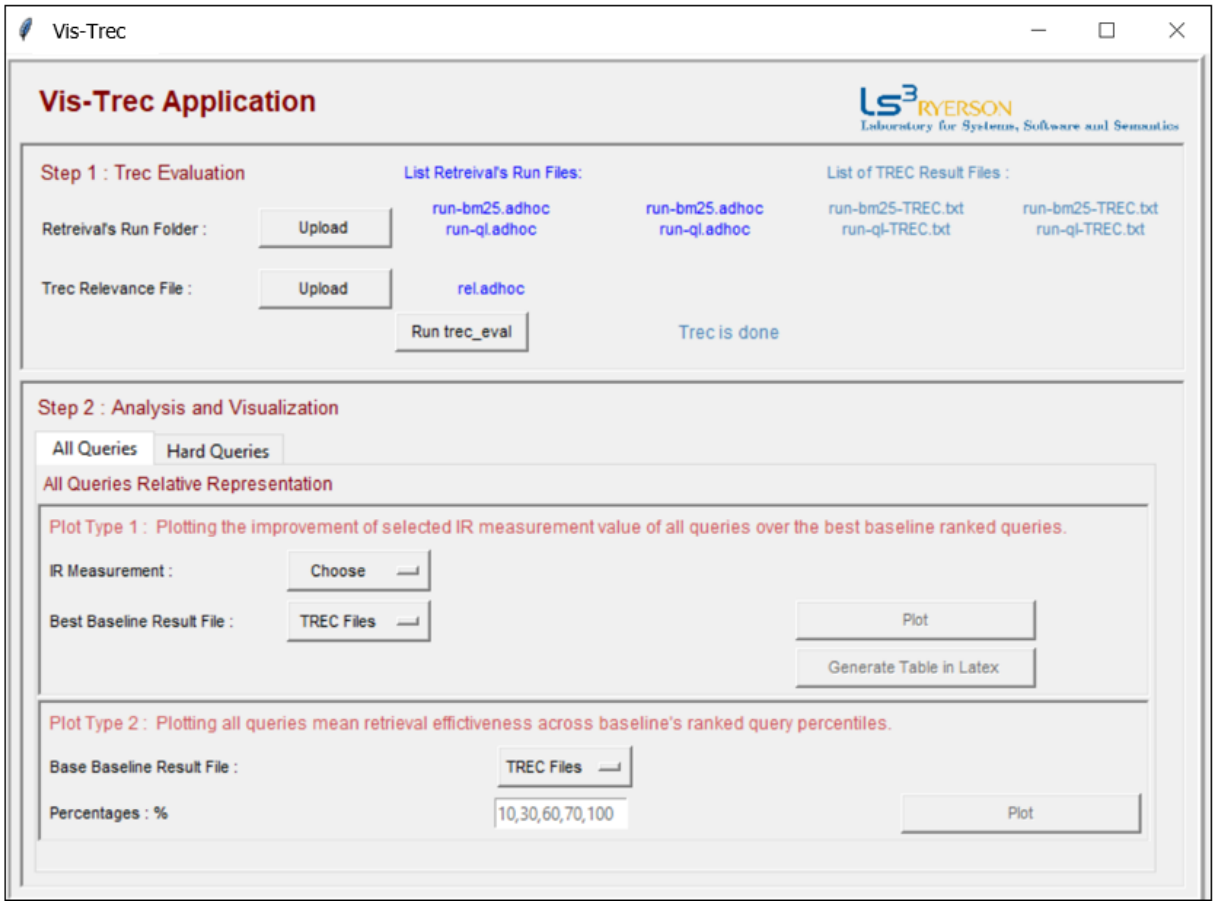

Figure. 4 Screenshot of the Vis-Trec Graphical User Interface for Working with All Available Queries.

[9] Christophe Van Gysel and Maarten de Rijke. 2018. Pytrec_eval: An Extremely Fast Python Interface to trec_eval. In *International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 873–876.

[10] Ellen M Voorhees. 2001. The philosophy of information retrieval evaluation. In *Workshop of the cross-language evaluation forum for european languages*. Springer, 355–370.