Judging the Judges: A Collection of LLM-Generated Relevance Judgements

Hossein A. Rahmani
University College London
London, UK
hossein.rahmani.22@ucl.ac.uk

Clemencia Siro
University of Amsterdam
Amsterdam, The Netherlands
c.n.siro@uva.nl

Mohammad Aliannejadi
University of Amsterdam
Amsterdam, The Netherlands
m.aliannejadi@uva.nl

Nick Craswell
Microsoft
Bellevue, US
nickcr@microsoft.com

Charles L. A. Clarke
University of Waterloo
Waterloo, Ontario, Canada
claclark@gmail.com

Guglielmo Faggioli
University of Padua
Padua, Italy
faggioli@dei.unipd.it

Bhaskar Mitra
Microsoft
Montréal, Canada
bmitra@microsoft.com

Paul Thomas
Microsoft
Adelaide, Australia
pathom@microsoft.com

Emine Yilmaz
University College London & Amazon
London, UK
emine.yilmaz@ucl.ac.uk

Abstract

Using Large Language Models (LLMs) for relevance assessments offers promising opportunities to improve Information Retrieval (IR), Natural Language Processing (NLP), and related fields. Indeed, LLMs hold the promise of allowing IR experimenters to build evaluation collections with a fraction of the manual human labor currently required. This could help with fresh topics on which there is still limited knowledge and could mitigate the challenges of evaluating ranking systems in low-resource scenarios, where it is challenging to find human annotators. Given the fast-paced recent developments in the domain, many questions concerning LLMs as assessors are yet to be answered. Among the aspects that require further investigation, we can list the impact of various components in a relevance judgment generation pipeline, such as the prompt used or the LLM chosen.

This paper benchmarks and reports on the results of a large-scale automatic relevance judgment evaluation, the *LLMJudge challenge* at SIGIR 2024, where different relevance assessment approaches were proposed. In detail, we release and benchmark 42 LLM-generated labels of the TREC 2023 Deep Learning track relevance judgments produced by eight international teams who participated in the challenge. Given their diverse nature, these automatically generated relevance judgments can help the community not only investigate systematic biases caused by LLMs but also explore the effectiveness of ensemble models, analyze the trade-offs between different models and human assessors, and advance methodologies for improving automated evaluation techniques. The released resource is available at the following link: https://llm4eval.github.io/LLMJudge-benchmark/.

ACM Reference Format:
Hossein A. Rahmani, Clemencia Siro, Mohammad Aliannejadi, Nick Craswell, Charles L. A. Clarke, Guglielmo Faggioli, Bhaskar Mitra, Paul Thomas, and Emine Yilmaz. 2018. Judging the Judges: A Collection of LLM-Generated Relevance Judgements. In . ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

The Cranfield paradigm has been the standard Information Retrieval (IR) evaluation methodology since the 1960s [6, 7]. This methodology requires three components to evaluate an IR system: a document corpus, topics (queries), and *corresponding relevance judgments* that indicate which documents are relevant to each topic.

In practice, an IR system processes the topics to retrieve relevant documents from the corpus, while the relevance judgments serve as the ground truth for measuring system effectiveness. Of these three components, the first two are relatively straightforward to acquire. The document corpus can be constructed to mirror the target domain of the IR system through various collection methods, such as web crawling, newspaper archives, or scientific literature databases. Similarly, topics can be manually handcrafted by non-experts or through the systematic analysis of query logs [22], ensuring they represent an appropriate sample of the expected query space.

The main challenge lies in obtaining relevance judgments. These judgments map topics to documents, specifically indicating how well each document satisfies the information need expressed in the topic. Creating complete relevance judgments requires significant human effort and careful quality control to ensure consistency and reliability [30].

Over time, three major approaches became the de facto standard to collect relevance judgments. The first approach, followed by the major evaluation campaigns such as TREC [17], NTCIR [18] and CLEF [24], is based on editorial assessments. In this case, professional assessors judge whether a document is relevant in response to the topic. While these judgments are often of very high quality,

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/XXXXXXX.XXXXXXX

they are also expensive to obtain in terms of time and cost [30]. A second strategy is based on employing crowd-assessors to annotate the documents. While crowd annotations are typically less expensive than editorial annotations, they are also qualitatively inferior. Crowd annotations often contain much more noise and errors. Thirdly, annotations can be obtained as implicit feedback from user - IR system interaction. These annotations are virtually free as they are based mostly on already available data (e.g., click logs) and can embed user-specific characteristics, such as their knowledge, tastes, and personal inclinations. Nevertheless, implicit feedback is also affected by noise, biases [16], and privacy problems [2]. In general, the types of annotations can be organized in a spectrum: on one side we have accurate and costly editorial judgments, in the middle we have labels produced by the crowd, on the other side, we find inexpensive but imprecise and biased implicit feedback.

Large Language Models (LLMs) have recently emerged as a promising fourth approach to gather relevance judgments [12, 20, 25, 34]. Initial experiments show that LLMs can achieve comparable performance to crowd workers on standard IR tasks [4] and potentially reduce annotation costs. The LLMJudge challenge [29] was organized as part of the LLM4Eval[1] workshop [27] at SIGIR 2024 as a shared task to study the effectiveness of using LLMs as annotation tools. While LLMs have shown to be effective annotation tools, several aspects are yet to be understood. For example, it is not clear the impact of changing the prompt, which biases are present in the LLM-generated relevance judgments, and if there is a risk of evaluation circularity [12]. Beyond these challenges, there is also a need to explore the effectiveness of ensemble models, examine the trade-offs between different LLM-based and human assessments, and develop more advanced methodologies to enhance automated evaluation techniques. We release the relevance annotations produced by the teams participating in the LLMJudge challenge, to help the community investigate these aspects linked to using LLM as automatic annotation tools.

Our contributions are the following:

- We release 42 pools of automatically generated relevance judgments produced by 8 different research teams that participated in the LLMJudge challenge.
- We confirm current observations about the state of the art, noticing that, while many approaches maintain ranking consistency, their absolute scoring tendencies differ, potentially introducing biases in evaluation.
- From the methodological perspective, we investigate several approaches that can be adopted to assess the effectiveness of an LLM-based relevance judgment process and provide a set of figures that will serve future researchers as baselines.

The remainder of this paper is organized as follows: Section 2 introduces the related works, including the challenges associated with automatically generated relevance judgments. Section 3 delineates the structure and describes the collection of the LLMJudge resource. Section 4 summaries submission runs of LLMJudge challenge. In Section 5, we analyze the dataset and provide some insights on the LLM-generated judgments. Finally, in Section 6, we draw our conclusion and outline our future work.

[1]https://llm4eval.github.io/

2 Related Work

Traditionally, an experimental IR collection includes three elements, a corpus, a set of topics, and the relevance judgments, defining which documents are relevant in response to the topics. Over the last 30 years, since the first TREC campaign [15], the most common strategy to obtain such relevance judgments has involved expert annotators, capable of providing the most accurate labels. The cost of this process can be partially reduced with pooling [9], but the monetary and temporal costs of building an IR experimental collection following this paradigm remain extremely high.

Automatic relevance judgment has recently received significant attention in the IR community. In earlier studies, Faggioli et al. [12] studied different levels of human and LLMs collaboration for automatic relevance judgment. They suggested the need for humans to support and collaborate with LLMs for a human-machine collaboration judgment. Thomas et al. [34] leverage LLMs capabilities in judgment at scale, in Microsoft Bing. They used real searcher feedback to build an LLM and prompt in a way that matches the small sample of searcher preferences. Their experiments show that LLMs can be as good as human annotators in indicating the best systems. They also comprehensively investigated various prompts and prompt features for the task and revealed that LLM performance on judgments can vary with simple paraphrases of prompts. Recently, Rahmani et al. [25] have studied fully synthetic test collection using LLMs. In their study, they generated synthetic queries and synthetic judgment to build a full synthetic test collation for retrieval evaluation. They have shown that LLMs can generate a synthetic test collection that results in system ordering performance similar to evaluation results obtained using the real test collection.

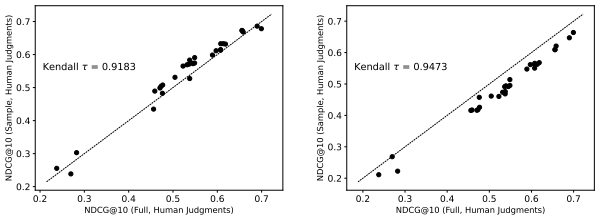
On a different line, Dietz [10] defines a LLM-based "autograding" approach. This evaluation strategy targets generated content that cannot be evaluated in a purely offline scenario and it consists of using a question bank as the evaluation test-bed. An LLM measures the effectiveness of the generative model in answering the questions, possibly with the supervision of a human. The autograding approach proposed by Dietz [10] includes an automatic passage evaluation whose task aligns with the one evaluated in LLMJudge.

2.1 Criticisms and Open Challenges

The use of LLMs as assessors comes with major bias risks and challenges that should not be neglected, especially considering the impact they might have in the development of IR evaluation.

Bias. First and most importantly, LLMs are affected by bias [3]. Their internal representation of the concepts is, by construction, conditioned on the context such concepts appear in [36]. Thus, depending on the underlying data, the LLM might form a biased notion of relevance that might reflect upon the relevance judgments generated by it. Quantifying the bias, identifying its source, and mitigating its consequences are still open issues that need to be addressed. We hope that the release of this collection will help the research community with the needed data to study how to deal with the bias in LLM-generated relevance judgments.

Circularity. A second source of concern when it comes to using LLMs as assessors relates to the risk of *circular evaluation* [12, 32]. For example, the same LLM might be used to generate relevance



(a) Dev set **(b) Test set**

Figure 1: Samples correlation with TREC 2023 DL full qrel

judgments and as a document ranker. This would induce a strong bias on the validity and generalizability of the relevance judgments.

Environmental Impact. An often hidden cost of the LLMs concerns their environmental impact in terms of energy utilization, carbon emissions [31, 37], and water consumption [38]. While LLMs might allow building collections at a fraction of the monetary and temporal cost, we should account for the environmental impact of such a process, limiting our reliance on "disposable" relevance judgments.

Vulnerability to Attacks and Adversarial Misuse. Parry et al. [23] and Alaofi et al. [1] illustrate the vulnerability of the LLMs to mischievous manipulations of the corpus. For example, Parry et al. [23] show that, by introducing keywords such as the term "relevant" in a document, it will more likely considered relevant by an LLM. Similar behavior is observed also by Alaofi et al. [1], who notice that by introducing the query on the document, more probably an LLM will consider the document relevant to such a query — even if the rest of the document is composed by random terms. More recently, Clarke and Dietz [5] show how, by properly crafting an adversarial run, it is possible to cheat an LLM used as an assessor. Clarke and Dietz [5] crafted a run following the same approach used by Upadhyay et al. [35] to pool the documents and build the LLM-generated relevance judgments used for TREC 2024 RAG. Such a run achieved consistently higher effectiveness under the fully automatic evaluation paradigm compared to its performance based on manual relevance judgments.

By releasing this collection of LLM-generated relevance judgments we want to foster the analysis and study of possible sources of biases and systematic errors, to mitigate them and allow for the development of more effective and robust future solutions that involve LLMs as tools to support the annotation process.

3 LLMJudge Resource

This section details how we designed the LLMJudge challenge task, the data construction process, and the evaluation metrics.

3.1 LLMJudge Task

The task of the LLMJudge challenge is, given the query and passage as input, how they are relevant. Similar to TREC 2023 Deep Learning track [8], we use *four-point scale* judgments to evaluate the relevance of the query to the passage as follows:

- **[3] Perfectly relevant**: The passage is dedicated to the query and contains the exact answer.
- **[2] Highly relevant**: The passage has some answers for the query, but the answer may be a bit unclear, or hidden amongst extraneous information.
- **[1] Related**: The passage seems related to the query but does not answer it.
- **[0] Irrelevant**: The passage has nothing to do with the query.

More specifically, the LLMJudge challenge is, by providing the datasets that include queries, passages, and query-passage files to participants, to ask LLMs to generate a score [0, 1, 2, 3] indicating the relevance of the query to the passage.

Table 1: Statistics of LLMJudge Dataset

	Dev	Test
# queries	25	25
# passage	7,224	4,414
# qrels	7,263	4,423
# irrelevant (0)	4,538	2,005
# related (1)	1,403	1,233
# highly relevant (2)	625	808
# perfectly relevant (3)	697	377

3.2 LLMJudge Data

The LLMJudge challenge dataset is built upon the passage retrieval task dataset of the TREC 2023 Deep Learning track[2] [8]. The TREC 2023 Deep Learning track qrel consists of 82 queries, including 51 real queries and 31 synthetic queries (13 generated by T5 and 18 generated by GPT-4). To create a dev and test set similar to the TREC 2023 Deep Learning track full qrel, we randomly sampled 15 queries from 51 real queries, 5 queries from T5 queries, and 5 queries from GPT-4 queries for each set. Figure 1 shows Kendall's τ correlation of the TREC 2023 Deep Learning track run submission on LLMJudge sampled dev and test sets with the TREC 2023 Deep Learning track full qrel. Table 1 shows the statistics of the LLMJudge challenge datasets. The test set is used for the generation of judgment by participants, while the development set could be used for few-shot or fine-tuning purposes.

3.3 Evaluation

We evaluate submission results on two different levels, the correlation of the judgments and the ranking correlation of systems evaluated using judgment submissions:

- **Label Correlation.** We use the automated evaluation metrics Cohen's Kappa (κ) and Krippendorff's Alpha (α) on human judgments and the judgments submitted by participants;
- **System Ranking Correlation.** We use Kendall's Tau (τ) and Spearman's rank (ρ) correlation to evaluate the system ordering of TREC 2023 Deep Learning Track [8] submitted systems on human judgments and participants' LLM-based judgments.

[2]https://microsoft.github.io/msmarco/TREC-Deep-Learning.html

We use `scikit-learn`[3] to compute Cohen's κ, Kendall's τ, Spearman's ρ. Krippendorff's α is also calculated using the Fast Krippendorff[4] Python package.

3.4 Publicly Available Resources

To facilitate research in the area we have made the LLMJudge dataset, sample prompt, quick starter for automatic judgment, submitted runs, prompts, codes for quick starting the evaluation, and more detailed results publicly available on the `LLMJudge` webpage at: https://llm4eval.github.io/LLMJudge-benchmark/.

4 Submitted Runs

We provide all submitted runs as a resource for future research and comparison. The submissions include 9 *baseline approaches* developed by the organizers and 33 *methods from participating teams*. Analysis of these submissions reveals several methodological directions in LLM-based relevance assessment, focusing on *prompting techniques, model adaptation, multi-phase evaluation, aggregation strategies, and classification-based refinement.*

Most submissions implement either *direct prompting* or *criteria decomposition pipelines. Direct prompting methods* range from simple *relevance scoring instructions* to *chain-of-thought reasoning*, where LLMs justify their judgments before assigning a score. Some approaches explore *zero-shot prompting*, while others incorporate *semantic label assignments, linguistic alignment, or multi-prompt aggregation* to improve consistency and reduce overestimation biases. Beyond prompting, some teams *fine-tune LLMs on relevance datasets*, including *TREC Deep Learning track qrels* and the *LLMJudge development set*, testing different *model sizes (8B vs. 70B)* to assess the impact of adaptation on evaluation performance. A subset of submissions structures evaluation into *multi-phase pipelines*, applying *binary filtering before graded scoring*, *question-based reasoning*, or *decision trees* to refine assessments. Other approaches decompose relevance into *specific dimensions* such as *exactness, coverage, topicality, and contextual fit*, or employ *nugget-based assessments* for more granular judgments. To enhance robustness, several methods *combine outputs from multiple prompts or models* using *multi-prompt averaging, binary-to-graded conversions, or conservative ensembling* to stabilize scores. Others treat *relevance assessment as a classification task*, extracting features from LLM outputs and training *Machine Learning classifiers* to refine final scores and improve alignment with human judgments.

Below we detail the baselines and a summary of the submitted runs. We also summarize the submission details in Table 2.

LLMJudge Baseline. The baseline judges provided by the LLMJudge challenge organizers serve as reference methods for evaluation. Three distinct approaches are proposed as baselines: `llmjudge-simple`, `llmjudge-cot`, and `llmjudge-thomas`. The `llmjudge-simple` method employs a straightforward prompt, instructing the model to directly provide a relevance judgment based on the query and passage. In contrast, `llmjudge-cot` adopts a chain-of-thought (CoT) approach, prompting the model to articulate its reasoning process before delivering a judgment. Lastly, `llmjudge-thomas` incorporates the prompt design introduced by [34], offering an alternative strategy for evaluation.

Table 2: LLMJudge challenge submissions details. Ensemble (Ens.) indicates if submissions combine multiple judges or use them as features to train a classifier for judgment. LR: Logistic Regression, ET: ExtraTrees, GaussianNB: Gaussian Naive Bayes are classifiers. If a submission used multiple prompts, we consider the more advanced one (CoT > Zero-Shot) in this table. FT: Fine-Tuning, N: Numerical, S: Semantic.

Submission ID	Model	Size	FT	Prompt	Label	Ens.
NISTRetrieval-instruct0	Llama-3-Instruct	8B	-	Zero-shot	N	-
NISTRetrieval-instruct1	Llama-3-Instruct	8B	-	Zero-shot	N	-
NISTRetrieval-instruct2	Llama-3-Instruct	8B	-	Zero-shot	N	-
NISTRetrieval-reason0	Llama-3-Instruct	8B	-	CoT	N	-
NISTRetrieval-reason1	Llama-3-Instruct	8B	-	CoT	N	-
NISTRetrieval-reason2	Llama-3-Instruct	8B	-	CoT	N	-
Olz-exp	GPT-4o	-	-	Zero-Shot	S	-
Olz-gpt4o	GPT-4o	-	-	CoT	S	-
Olz-halfbin	Llama-3-Instruct	8B	-	CoT	S + N	LR
Olz-somebin	Llama-3-Instruct	8B	-	CoT	S + N	LR
Olz-multiprompt	Llama-3-Instruct	8B	-	CoT	S + N	✓
RMITIR-GPT4o	GPT-4o	-	-	Zero-Shot	N	✓
RMITIR-llama38b	Llama-3-Instruct	8B	-	Zero-Shot	N	✓
RMITIR-llama70B	Llama-3-Instruct	70B	-	Zero-Shot	N	✓
TREMA-4prompts	Llama-3-Instruct	8B	-	Zero-Shot	N	-
TREMA-CoT	Llama-3-Instruct	8B	-	CoT	N	-
TREMA-all	ChatGPT-3.5/FlanT5-Large	783M	-	Few-Shot	N	ET
TREMA-direct	ChatGPT-3.5/FlanT5-Large	783M	-	Few-Shot	N	ET
TREMA-naiveBdecompose	ChatGPT-3.5/FlanT5-Large	783M	-	Zero-Shot	N	GNB
TREMA-nuggets	ChatGPT-3.5/FlanT5-Large	783M	-	Zero-Shot	N	ET
TREMA-other	ChatGPT-3.5/FlanT5-Large	783M	-	Zero-Shot	N	-
TREMA-questions	ChatGPT-3.5/FlanT5-Large	783M	-	Zero-Shot	N	ET
TREMA-rubric0	ChatGPT-3.5/FlanT5-Large	783M	-	Zero-Shot	N	-
TREMA-sumdecompose	Llama-3-Instruct	8B	-	Zero-Shot	N	-
h2oloo-fewself	GPT-4o	-	-	Few-Shot	N	-
h2oloo-zeroshot1	Llama-3-Instruct	8B	✓	Zero-Shot	N	-
h2oloo-zeroshot2	Llama-3-Instruct	8B	✓	Zero-Shot	N	-
llmjudge-cot1	GPT-3.5-turbo	-	-	CoT	N	-
llmjudge-cot2	GPT-3.5-turbo-16k	-	-	CoT	N	-
llmjudge-cot3	GPT-4-32k	-	-	CoT	N	-
llmjudge-simple1	GPT-3.5-turbo	-	-	Zero-Shot	N	-
llmjudge-simple2	GPT-3.5-turbo-16k	-	-	Zero-Shot	N	-
llmjudge-simple3	GPT-4-32k	-	-	Zero-shot	N	-
llmjudge-thomas1	GPT-3.5-turbo	-	-	Zero-Shot	N	-
llmjudge-thomas2	GPT-3.5-turbo-16k	-	-	Zero-Shot	N	-
llmjudge-thomas3	GPT-4-32k	-	-	Zero-Shot	N	-
prophet-setting1	Llama-3-Instruct	8B	✓	Zero-Shot	S	-
prophet-setting2	Llama-3-Instruct	8B	✓	Zero-Shot	S	-
prophet-setting4	Llama-3-Instruct	8B	✓	Zero-Shot	S	-
willia-umbrela1	GPT-4o	-	-	Zero-Shot	N	-
willia-umbrela2	GPT-4o	-	-	Zero-Shot	S	-
willia-umbrela3	GPT-4o	-	-	Zero-Shot	S + N	✓

NISTRetrieval-instruct. This is a submission from NIST which has three different variants, namely, `NISTRetrieval-instruct0`, `NISTRetrieval-instruct1`, and `NISTRetrieval-instruct2` that aims to investigate the reproducibility of the method proposed by Thomas et al. [34] and the reproducibility capabilities of LLMs when we used them for automatic relevance judgment.

NISTRetrieval-reason. Similar to `NISTRetrieval-instruct`, this NIST submission includes three related methods – `NISTRetrieval-reason0`, `NISTRetrieval-reason1`, and `NISTRetrieval-reason2`. The team observed that prompting LLMs to provide reasoning across various tasks could improve response quality. To examine whether this approach could also enhance relevance judgment, they modified the prompt from Thomas et al. [34] to allow the LLM to generate reasoning. These three runs were included to assess the reproducibility capabilities of LLMs when used for evaluation.

[3] https://scikit-learn.org/stable/index.html
[4] https://github.com/pln-fing-udelar/fast-krippendorff

Prophet-setting. This method builds on the idea of fine-tuning an LLM with different available datasets for automatic relevance judgment, as described in [21][5]. Specifically, they fine-tuned `Llama-3-8B` under three different settings, training the model for five epochs in each. These settings include: `Prophet-setting1`, fine-tuned on the LLMJudge development set; `Prophet-setting2`, fine-tuned on the qrels of TREC-DL 2019, 2020, and 2021; and `Prophet-setting4`, which combines fine-tuning on the qrels of TREC-DL 2019, 2020, and 2021 with the LLMJudge development set.

William-umbrela1. This approach is zero-shot prompting the LLM to produce relevance assessments. They used UMBRELA [35] to generate relevance judgments using the prompting technique suggested by Thomas et al. [34]. The team mentioned that "*I tried many different approaches, but I did not manage to find anything that really seemed to consistently improve on zero-shot. It seemed like this dataset may have been harder and/or noisier than others referenced in the literature – on my development set it was hard to get* > 0.3 *Cohen's* κ*, whereas the literature mentions values of* 0.4 *up to* 0.6 *even.*".

William-umbrela2. The main idea of this method is to take the approach from the UMBRELA [35] – zero-shot prompting technique from Thomas et al. [34], but to see if the performance would be improved by asking the model to output semantic labels (i.e., *Irrelevant*, *Related*, *Highly relevant*, *Perfectly relevant*), rather than a numerical score (i.e., 0, 1, 2, 3).

William-umbrela3. This method is an ensemble of `William-umbrela1` and `William-umbrela2` approaches by taking the *min*. The team mentioned that "*The logic behind using min as an aggregator is that in this dataset, it pays to be conservative in the rating. They also said that on a subset of the training data that they held out for testing, this ensembling approach outperformed either of the two other approaches (i.e., William-umbrela1 and William-umbrela2)*".

H2oloo-fewself. This method uses the best prompt proposed by Thomas et al. [34] to instruct GPT-4o. It incorporates few-shot examples to guide the model in distinguishing between relevant labels effectively.

H2oloo-zeroshot1. This method fined-tuned a Llama-8B using the TREC DL 2019 to 2022 qrels for relevance judgment prediction.

H2oloo-zeroshot2. This method fined-tuned a Llama-8B using the TREC DL 2019 to 2022 qrels and the LLMJudge dev set qrel for relevance judgment prediction.

Olz-gpt4o. This method uses a simple prompt where they just ask for the relevance judgment without any special techniques. The idea is to see how models can solve relevance judgment tasks without considering any particular prompting or fine-tuning techniques. The primary goal is to assess whether a low-effort prompt could reliably derive relevance labels from LLMs that are practically usable.

Olz-exp. This method is similar to `Olz-gpt4o` but they also asked LLM to reason its judgment as part of the evaluation.

Olz-halfbin. This method leverages Llama-3 models with $8B$ and $70B$ parameters to assess document relevance using nine distinct prompts. These prompts are divided into two categories: four *graded relevance prompts*, which instruct the model to assign a score from 0 to 3 with slight instruction variations, and five *binary relevance prompts*, which require binary judgments with different definitions of relevance. Both model variants generate outputs for all nine prompts. These outputs serve as features for training a logistic regression classifier, which produces the final graded labels. Training is conducted using labels generated by GPT-4o (via the `Olz-gpt4o` method) rather than the standard development set annotations, based on the assumption that the development and test set labels may have been derived using different methods. Analyzing these discrepancies, the team found GPT-4o's judgments more aligned with their expectations, leading to its adoption as the primary reference for training.

Olz-somebin. The procedure of this method is identical to the `Olz-halfbin` method, except the logistic regression classifier was trained on the provided development set labels instead of those generated by GPT-4o (using `Olz-gpt4o` method).

Olz-multiprompt. This method, instead of using a classifier like `Olz-halfbin` and `Olz-somebin`, directly aggregated the relevance judgments by averaging. The binary labels were first scaled by multiplying them by three (to convert them into 0 or 3). Then a simple average was calculated across the nine prompts and rounded on a scale of 0 to 3, and the resulting value served as the final graded label.

RMIT-IR. This submission introduces three relevance assessors, `RMITIR-GPT4o`, `RMITIR-llama38b`, and `RMITIR-llama70B`. The proposed approach begins by having the LLM provide a binary relevance judgment to filter out irrelevant queries and improve irrelevance filtering. Next, three scores are generated, and averaged, and the result is rounded to produce the final score. The method was tested using three different LLMs: GPT-4o (`RMITIR-GPT4o`), Llama3-8B (`RMITIR-llama38b`), and Llama3-70B (`RMITIR-llama70B`). The team noted that "*GPT-4o appears to be the best-performing model based on our experiences*."

TREMA-4prompt. This method evaluates passage relevance by decomposing it into four specific criteria: exactness (how precisely the passage answers the query), coverage (proportion of content discussing the query), topicality (subject alignment between passage and query), and contextual fit (presence of relevant background). The evaluation follows a two-phase process where each criterion is first assessed independently and then combined through a final prompt to determine the overall relevance label. Full details of the criteria and rationale are provided in [13].

TREMA-CoT. This method implements a chain-of-thought evaluation process inspired by Sun et al. [33]. The approach consists of three phases: First, the LLM makes a binary relevance judgment (yes/no) of the passage. Based on this judgment, different relevance criteria are evaluated in the second phase - for relevant passages, exactness and coverage are assessed, while non-relevant passages are evaluated on contextual fit and topicality (all scored 0-3). In the final phase, these scores determine the overall relevance label: relevant passages receive labels 2-3 based on exactness and coverage scores, while non-relevant passages receive labels 0-1 based on contextual fit and topicality assessment.

[5]Code is available at https://github.com/ChuanMeng/QPP-GenRE

TREMA-other. This approach investigates whether aligning the linguistic styles of queries and passages can enhance relevance judgments. In the first phase, the LLM generates a query-like representation for each passage, designed to match the query's linguistic style and length. This generated query serves as a summary of the passage's content, formatted in a way that aligns with typical query phrasing. In the second phase, the LLM evaluates the similarity between the original query and the generated query on a scale from 0 to 3, corresponding to the relevance labeling system. Higher similarity scores indicate a stronger alignment between the passage's content and the query's intent. This method integrates linguistic style alignment with content relevance to improve relevance labeling.

TREMA-sumdecompose. This method consists of two phases. Phase one is identical to the `TREMA-4prompt` method, where the "relevance" is decomposed into four criteria, leading to four criteria-specific grades. In Phase Two, the individual grades from Phase One are summed to produce a total grade. Based on this total, a final relevance label between 0 and 3 is assigned to each query-passage pair: a total grade of 10-12 yields a relevance label of 3, 7-9 yields a relevance label of 2, 5-6 yields a relevance label of 1, and 0-4 yields a relevance label of 0.

TREMA-naiveBdecompose. This method consists of two phases. Phase one is identical to the `TREMA-4prompt` method, where the "relevance" is decomposed into four criteria, leading to four criteria-specific grades. In phase two, these decomposed grades are aggregated into a final relevance label using a Gaussian Naive Bayes model, implemented with Scikit-learn's GaussianNB() classifier. The model is trained on the decomposed feature grades and then predicts the relevance label for each passage.

TREMA-rubric0. This method is based on the RUBRIC Autograder Workbench [11]. This method defines the relevance of the query via 10 open-ended questions. The questions are generated using the ChatGPT 3.5 model. Each passage is scanned whether it is possible to answer each of the questions (and how well), which is captured as a grade. They use the FLAN-T5-large LLM from Huggingface to grade the answerability from 0 (worst) to 5 (best). Details and prompts are available in the Workbench benchmark [11]. The grades are mapped to relevance labels by a heuristic mapping on the second-highest grade achieved on any of the questions. Grade 5 is mapped to relevance label 3, grade 4 is mapped to label 1 and all other grades are mapped to label 0. This was the best manual mapping on the dev set [14].

TREMA-questions. Same question and grading as in `TREMA-rubric0`, but uses a more elaborate calibration for converting grades to relevance labels, based on `scikit-learn`'s ExtraTrees classifier. The classifier is based on features that include ranked grades for each question (sorted in descending order), ranked question difficulty (based on average grades across the pool), and counts of correct answers at various grade thresholds (e.g., number of answers graded 5, 4 or better, etc.). Each of these features is encoded using both one-hot and numerical representations to capture detailed information about question-based relevance. The classifier is trained on the dev set.

Table 3: Judgment and system ranking correlation of LLM-Judge submissions. κ: Cohen's Kappa, α: Krippendorff's alpha, τ: Kendall's Tau, ρ: Spearman's rank correlation. The best results per column are denoted in bold and the second best results are denoted in *italic*.

Submission ID	κ	α	τ	ρ
NISTRetrieval-instruct0	0.1877	0.3819	0.9440	0.9907
NISTRetrieval-instruct1	0.1874	0.3812	0.9440	0.9907
NISTRetrieval-instruct2	0.1880	0.3821	0.9440	0.9907
NISTRetrieval-reason0	0.1844	0.3874	0.9052	0.9810
NISTRetrieval-reason1	0.1845	0.3872	0.9009	0.9802
NISTRetrieval-reason2	0.1838	0.3874	0.9052	0.9810
Olz-exp	0.2519	0.4701	0.9009	0.9819
Olz-gpt4o	0.2625	0.5020	0.8793	0.9758
Olz-halfbin	0.2064	0.4536	0.9085	0.9830
Olz-multiprompt	0.2445	0.4551	0.9267	0.9867
Olz-somebin	0.2109	0.4471	0.9042	0.9822
RMITIR-GPT4o	0.2388	0.4108	0.8966	0.9798
RMITIR-llama38b	0.2006	0.3873	0.8879	0.9758
RMITIR-llama70B	0.2654	0.4873	0.9353	0.9883
TREMA-4prompts	0.1829	0.2888	*0.9483*	**0.9919**
TREMA-CoT	0.1961	0.3852	0.8956	0.9799
TREMA-all	0.1471	0.3855	0.9138	0.9863
TREMA-direct	0.1742	0.3729	0.9009	0.9819
TREMA-naiveBdecompose	0.1741	0.3579	0.9128	0.9838
TREMA-nuggets	0.0604	0.1691	0.8664	0.9718
TREMA-other	0.1408	0.2712	0.8276	0.9447
TREMA-questions	0.1137	0.3148	0.9095	0.9839
TREMA-rubric0	0.0779	0.1036	0.8276	0.9544
TREMA-sumdecompose	0.2088	0.3926	0.9300	0.9870
h2oloo-fewself	0.2774	**0.4958**	0.9085	0.9822
h2oloo-zeroshot1	*0.2817*	0.4812	0.9181	0.9827
h2oloo-zeroshot2	0.2589	0.3898	0.8353	0.9604
llmjudge-cot1	0.1284	0.3218	0.9267	0.9871
llmjudge-cot2	0.1560	0.3263	0.9267	0.9875
llmjudge-cot3	0.2271	0.4870	0.9267	0.9851
llmjudge-simple1	0.0754	0.2808	0.9181	0.9863
llmjudge-simple2	0.1327	0.3672	0.8966	0.9790
llmjudge-simple3	0.2110	0.4642	0.9052	0.9810
llmjudge-thomas1	0.1236	0.3207	0.8664	0.9689
llmjudge-thomas2	0.1723	0.3853	0.8793	0.9750
llmjudge-thomas3	0.2293	0.4877	0.9181	0.9867
prophet-setting1	0.1823	0.4069	0.9042	0.9826
prophet-setting2	0.1757	0.3144	**0.9516**	*0.9914*
prophet-setting4	0.1471	0.1623	0.8568	0.9608
willia-umbrela1	**0.2863**	*0.4918*	0.9009	0.9806
willia-umbrela2	0.2688	0.4556	0.8870	0.9769
willia-umbrela3	0.2741	0.4535	0.8707	0.9730

TREMA-nuggets. Same approach as `TREMA-questions`, but uses 10 open-ended key fact nuggets instead of questions, along with an adapted prompt that assesses whether key facts are mentioned in the passage. The same ExtraTrees classifier with the same features is used for converting grades into relevance labels.

TREMA-direct. This approach focuses exclusively on features of direct relevance labeling methods, which instruct an LLM to judge

whether a passage is relevant for a query, using a variety of prompts from Sun et al. [33], Faggioli et al. [12], and HELM [19]. The model excludes question-based and nugget-based features, simplifying its input to focus solely on the predictive power of direct labeling. The relevance labels are obtained with an ExtraTrees classifier trained on the dev set. Features include binary or multi-class predictions from labeling approaches. Each label is encoded using both one-hot and numerical encodings to capture both categorical and ordinal aspects of the predictions. This approach is computationally lighter than TREMA-all and serves as a baseline to evaluate how well direct relevance labels alone can predict passage relevance.

TREMA-all. This approach incorporates all features from `TREMA-questions`, `TREMA-nuggets`, and `TREMA-direct` approaches via a single ExtraTrees classifier that is trained on the dev set.

5 Results

5.1 Descriptive Statistics

For the LLMJudge challenge, we received a total of 42 submissions (i.e., the 42 labelers) from 8 teams. Of these, 9 runs were from the organizers, who contributed a range of baselines; these are included in the statistics. Among the submissions, 5 utilized fine-tuning for relevance judgment, while the rest relied on various prompting techniques. Moreover, 16 submissions are based on proprietary LLMs, whereas 26 used open-source LLMs.

5.2 Methodological Comparison

Here we provide an overview analysis of the submissions based on different methodological directions they applied for their LLM-based relevance assessments. Table 3 presents the results of submissions at the label and system ranking correlation. Comparing submissions concerning the models they used, we can see that while `willia-umbrela1` (GPT-4o) achieves the best Cohen's κ, `h2oloo-zeroshot1` (Llama3-8B) ranked second by only 1.61% differences. More interestingly, when we compare the system ranking correlation (i.e., Kendall's τ) of submissions, the best non-fine-tuned method is `TREMA-4prompts` which uses Llama3-8B. Previous studies [28, 34] have shown the importance of the prompting techniques for automatic relevance judgments, and analyzing the LLMJudge submission results confirms the importance of the effect of prompt engineering. For instance, `TREMA-4prompts` uses the criteria decomposition technique by breaking down the concept of relevance into various criteria and generating the relevance label by asking the model about the specified criteria. Few submissions used fine-tuning, however, their results show that fine-tuning can lead to the highest correlation. For instance, `h2oloo-fewself` achieved the highest Krippendorff's α (0.4958, the best among all methods) and `prophet-setting2` is the best-performing submission considering Kendall's τ. This confirms that fine-tuning can significantly enhance agreement with human judgments. Submissions that included both numerical and semantic labels tend to perform consistently across all evaluation metrics compared to those using only numerical labels. For example, submissions from the "Olz" team rank among the comparable submissions across all four evaluation metrics. In the following, we provide a more detailed discussion and analysis.

Table 4: Cohen's κ correlation in 4-point scale agreement and difference binarize the judgment scale. Judgment levels to the left of the pipe are considered irrelevant, while those to the right are considered relevant.

Submission ID	4-point	0|123	01|23	012|3
NISTRetrieval-instruct0	0.1877	0.3116	0.3021	0.0000
NISTRetrieval-instruct1	0.1874	0.3106	0.3021	0.0000
NISTRetrieval-instruct2	0.1880	0.3126	0.3013	0.0000
NISTRetrieval-reason0	0.1844	0.2911	0.3390	0.0097
NISTRetrieval-reason1	0.1845	0.2906	0.3394	0.0097
NISTRetrieval-reason2	0.1838	0.2902	0.3397	0.0097
Olz-exp	0.2519	0.3997	0.3577	0.2936
Olz-gpt4o	0.2625	**0.4228**	0.3657	0.3066
Olz-halfbin	0.2064	0.4008	0.2587	0.2449
Olz-multiprompt	0.2445	0.3764	0.3934	0.2150
Olz-somebin	0.2109	0.3854	0.3883	0.1137
RMITIR-GPT4o	0.2388	0.3499	0.3961	0.2580
RMITIR-llama38b	0.2006	0.3280	0.3194	0.1344
RMITIR-llama70B	0.2654	*0.4166*	0.3916	0.2843
TREMA-4prompts	0.1829	0.3022	0.2697	0.1664
TREMA-CoT	0.1961	0.3181	0.3208	0.1836
TREMA-all	0.1471	0.3244	0.2978	0.0717
TREMA-direct	0.1742	0.3205	0.3462	0.1763
TREMA-naiveBdecompose	0.1741	0.3085	0.2916	0.0153
TREMA-nuggets	0.0604	0.1505	0.0992	-0.0077
TREMA-other	0.1408	0.2740	0.2015	0.1411
TREMA-questions	0.1137	0.2636	0.2876	0.0441
TREMA-rubric0	0.0779	0.1714	0.0308	0.0369
TREMA-sumdecompose	0.2088	0.3228	0.3512	0.2047
h2oloo-fewself	0.2774	0.4172	0.4280	0.3048
h2oloo-zeroshot1	0.2817	0.4094	0.3901	0.3084
h2oloo-zeroshot2	0.2589	0.3691	0.3278	0.2789
llmjudge-cot1	0.1284	0.2219	0.2833	0.1287
llmjudge-cot2	0.1560	0.2507	0.2944	0.2048
llmjudge-cot3	0.2271	0.3856	0.3978	0.2335
llmjudge-simple1	0.0754	0.1278	0.2880	0.1582
llmjudge-simple2	0.1327	0.2349	0.3241	0.2004
llmjudge-simple3	0.2110	0.3590	0.3972	0.2157
llmjudge-thomas1	0.1236	0.2087	0.2843	0.1802
llmjudge-thomas2	0.1723	0.2944	0.3043	0.2267
llmjudge-thomas3	0.2293	0.3910	0.3947	0.2438
prophet-setting1	0.1823	0.3502	0.2903	0.1677
prophet-setting2	0.1757	0.3102	0.2382	0.0284
prophet-setting4	0.1471	0.2375	0.1409	0.0371
willia-umbrela1	0.2863	0.4161	**0.3985**	0.3145
willia-umbrela2	0.2688	0.4109	0.3421	0.3194
willia-umbrela3	0.2741	0.4114	0.3447	**0.3215**

5.3 Overall Results

The results of the LLMJudge challenge, as presented in Table 3, reveal significant variability in performance across the evaluated metrics, including Cohen's Kappa (κ), Krippendorff's Alpha (α), Kendall's Tau (τ), and Spearman's Rho (ρ). Submissions such as `h2oloo-fewself`, `h2oloo-zeroshot1`, and `willia-umbrela1` emerge as top performers, achieving κ values above 0.27 and α values around 0.49, indicative of strong agreement and reliability. These methods also demonstrate robust ranking capabilities, with τ and ρ

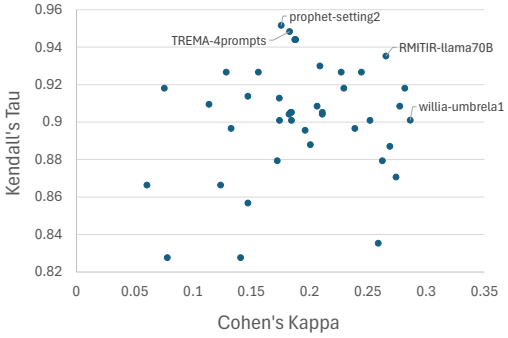
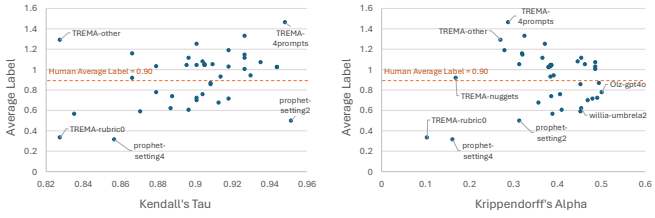


Figure 2: Cohen's κ vs. Kendall's τ



(a) Avg. Label vs. Kendall τ **(b) Avg. Label vs. Krippendorff's α**

Figure 3: Average Labels

values exceeding 0.9. This combination of high agreement and ranking correlation suggests that these approaches are well-suited for relevance judgment tasks. In contrast, submissions like `TREMA-nuggets` and `TREMA-rubric0` perform poorly, with $\kappa < 0.1$ and $\alpha < 0.2$, reflecting low agreement and reliability. Despite their low scores on agreement metrics, some of these models maintain moderate ranking correlations, indicating limited but specific utility in ranking-focused scenarios.

A comparison of group trends highlights the strengths and weaknesses of different methodological approaches. For instance, `NISTRetrieval` submissions consistently achieve high τ and ρ values (> 0.9), reflecting strong ranking performance, yet their lower κ (~0.18) and α (~0.38) suggest limited alignment with human relevance judgments. In contrast, methods like `Olz-multiprompt` and `h2oloo-fewself` demonstrate comparable performance across all metrics. `Olz-multiprompt` leverages outputs from multiple prompts and `h2oloo-fewself` incorporates few-shot examples using a proprietary model, GPT4o, these approaches effectively mitigate individual model biases and enhance both reliability and agreement. On the other hand, single-model approaches, such as `TREMA-direct` and `llmjudge-simple1`, exhibit limited performance, underscoring the challenges faced by individual models in capturing the complexity of relevance judgment tasks.

5.4 Label Correlation (Cohen's κ)

Table 4 presents Cohen's κ values for various submissions, providing insights into the agreement between relevance judgments under different granularity levels: 4-point, 0|123, 01|23, and 012|3. Across all groupings, there is noticeable variability in κ scores, highlighting differences in consistency among submissions. Submissions like `h2oloo-fewsel` and `h2oloo-zeroshot1` demonstrate relatively high agreement across all groupings, particularly excelling in the 4-point and binary (0|123) categories, with κ values exceeding 0.27 in most cases. In contrast, submissions such as `TREMA-nuggets` and `TREMA-rubric0` exhibit significantly lower agreement, with κ values as low as 0.0604 and 0.0779 on the 4-point scale, reflecting limited reliability in their judgments.

In particular, coarse-grained groupings like 0|123 tend to produce higher κ values than finer groupings like the 4-point scale, suggesting that systems or annotators achieve better consistency when relevance levels are aggregated. However, the 012|3 grouping, which isolates the highest relevance level, introduces greater variability, with some systems such as `willia-umbrela1` performing well, while others struggle to maintain consistency. These findings emphasize the importance of evaluation granularity in understanding system reliability and identifying approaches with stable performance across diverse grouping strategies.

5.5 Cohen's κ vs. Kendall's τ

Figure 2 shows the performance of submitted runs on the `LLMJudge` test set. The x-axis represents Cohen's κ, and the y-axis shows the submission agreement on system order. Submissions exhibit low variability in Kendall's τ but greater variability in Cohen's κ. Most submissions cluster within a narrow range of τ values, indicating consistent system rankings but more variation in inter-rater reliability, as measured by Cohen's κ. This suggests that while submissions generally agree on rankings, their exact labels are less consistent, leading to the observed variability in κ.

5.6 Average Label Comparison

Figure 3 illustrates the relationship between ranking correlation metrics and the average label assigned by different evaluation methods for NDCG@10. The orange dashed line represents the human average label (0.90), serving as a baseline for comparison. In Figure 3a, most methods exhibit strong agreement in ranking order, with values generally above 0.85. However, the assigned average labels vary significantly, with some methods, such as `TREMA-other` and `TREMA-4prompts`, assigning scores notably above the human baseline, while others, like `TREMA-rubric0` and `prophet-setting4`, produce lower scores. These variations indicate that while many approaches maintain ranking consistency, their absolute scoring tendencies differ, potentially introducing biases in evaluation.

Figure 3b provides further insight into inter-method agreement, capturing not just ranking order but also overall consistency in score distributions. Here, we observe a wider range of correlation values, with some methods achieving moderate agreement (e.g., `Olz-gpt4o` and `willia-umbrella2`) while others, such as `TREMA-rubric0` and `prophet-setting4`, show very low agreement and lower assigned scores. Notably, `TREMA-4prompts` and `TREMA-other` again stand out with higher average labels, but their variability suggests differences in how they align with human judgments. These findings emphasize the importance of calibration when aggregating synthetic judgments, as different methods may systematically overestimate or underestimate relevance scores despite high-ranking agreement.

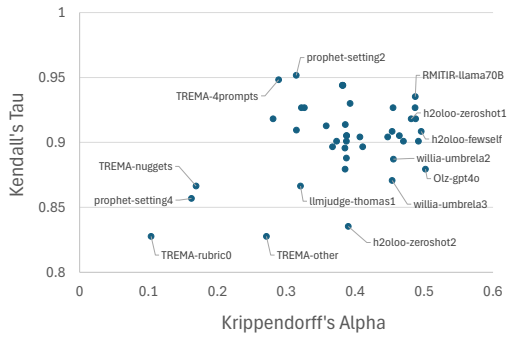


Figure 4: Krippendorff's α vs. Kendall's τ

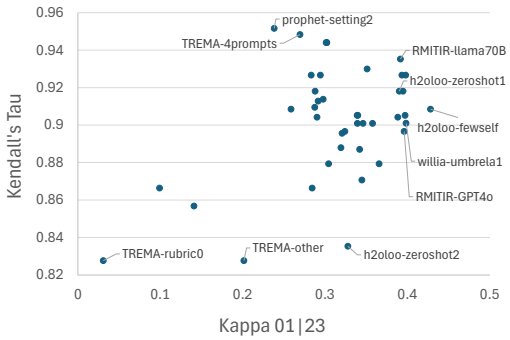


Figure 5: Binarized Cohen's κ vs. Kendall's τ

5.7 Krippendorff's α vs. Kendall's τ

Figure 4 shows the relation between Krippendorff's α and Kendall's τ, highlighting the agreement of different models with human judgments. Higher Krippendorff's α generally corresponds to better Kendall's τ, but with notable variance. Models like `prophet-setting2` and `TREMA-4prompts` achieve strong ranking consistency, while `TREMA-rubric0` and `TREMA-other` show weaker agreement. Proprietary models such as `h2oloo-fewself` and `Olz-gpt4o` perform well in both metrics, whereas some open-source models are more dispersed.

5.8 Binarized Cohen's κ vs. Kendall's τ

Figure 5 compares binary agreement (Cohen's κ 01|23) with Kendall's τ across models. Higher kappa values tend to align with stronger ranking consistency, as seen with `prophet-setting2`, `TREMA-4prompts`, and `RMITIR-llama70B`. However, some models, like `TREMA-rubric0` and `TREMA-other`, show weak agreement despite moderate ranking correlations. Proprietary models, including `RMITIR-GPT4o` and `h2oloo-fewself`, perform competitively, suggesting that both fine-tuning and prompting strategies impact these relationships.

5.9 LLMJudge Resource Use Cases

The LLMJudge benchmark can be considered as a resource for evaluating the reliability of LLM-generated relevance judgments across different settings. For instance, JudgeBlender [28] which is a framework that aggregates evaluations from multiple smaller models to enhance the robustness of relevance assessments recently leveraged LLMJudge both as a baseline for comparison and as a tool for analyzing the variability and bias of ensemble-based judgment aggregation methods.

6 Conclusion

We introduce the LLMJudge resource, which builds upon the foundations established by the LLMJudge challenge [29] at the LLM4Eval workshop [26, 27] co-located with SIGIR 2024. This resource provides a benchmark for evaluating the effectiveness of LLMs in an automatic relevance judgment task, helping comparisons across different models and prompting strategies. In this paper, we detail the 42 sets of relevance judgments for the TREC 2023 Deep Learning track submitted to the LLMJudge challenge by 8 different international teams. We release this collection to serve multiple purposes. First, they can be used as a comparison for future LLM-based relevance assessments, allowing research teams that did not participate in the challenge to compare their approaches as well. Secondly, this resource can serve as a tool to determine or help empirically study the presence of systematic biases in LLM-generated relevance judgments, impacting a large body of research in the community. Among the submitted sets of relevance judgments, 5 employ fine-tuning, and their results show that fine-tuning can lead to the highest correlation. Furthermore, 16 submissions are based on proprietary LLMs and 26 on open-source LLMs. Our analyses show that the latter tend to be more stable, while the former are affected by higher variability. Methodologically, we provide in this paper a set of strategies to compare multiple automatically-generated relevance assessments that can serve future practitioners in determining the effectiveness of new LLMs as assessors. In future work, we plan to investigate how LLM can be used to generate relevance judgment in a nugget-based evaluation scenario and extend the analysis to fully automatic collections, that include automatically generated queries and documents.

Acknowledgments

The challenge is organized as a joint effort by the University College London, Microsoft, the University of Amsterdam, the University of Waterloo, and the University of Padua. The views expressed in the content are solely those of the authors and do not necessarily reflect the views or endorsements of their employers and/or sponsors. This research is supported by the Engineering and Physical Sciences Research Council [EP/S021566/1], CAMEO, PRIN 2022 n. 2022ZLL7MW and by the Dreams Lab, a collaboration between Huawei Finland, the University of Amsterdam, and the Vrije Universiteit Amsterdam.

References

[1] Marwah Alaofi, Paul Thomas, Falk Scholer, and Mark Sanderson. 2024. LLMs can be Fooled into Labelling a Document as Relevant: best café near me; this paper is perfectly relevant. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region, SIGIR-AP 2024, Tokyo, Japan, December 9-12, 2024.* ACM, 32–41. https://doi.org/10.1145/3673791.3698431

[2] Pol Mac Aonghusa and Douglas J. Leith. 2016. Don't Let Google Know I'm Lonely. *ACM Trans. Priv. Secur.* 19, 1 (2016), 3:1–3:25. https://doi.org/10.1145/2937754

[3] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?. In *FAccT '21: 2021 ACM Conference on Fairness, Accountability, and*

Transparency, Virtual Event / Toronto, Canada, March 3-10, 2021. ACM, 610–623. https://doi.org/10.1145/3442188.3445922

[4] Roi Blanco, Harry Halpin, Daniel M. Herzig, Peter Mika, Jeffrey Pound, Henry S. Thompson, and Duc Thanh Tran. 2011. Repeatable and reliable search system evaluation using crowdsourcing. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011*. ACM, 923–932. https://doi.org/10.1145/2009916.2010039

[5] Charles L. A. Clarke and Laura Dietz. 2024. LLM-based relevance assessment still can't replace human relevance assessment. *CoRR* abs/2412.17156 (2024). https://doi.org/10.48550/ARXIV.2412.17156 arXiv:2412.17156

[6] Cyril Cleverdon. 1967. *The Cranfield tests on index language devices*. Vol. 19. Emerald, San Francisco, CA, USA, 173–194. Issue 6. https://doi.org/doi:10.1108/eb050097

[7] Cyril W. Cleverdon. 1960. The Aslib Cranfield Research Project on the Comparative Efficiency of Indexing Systems. https://api.semanticscholar.org/CorpusID:60470177

[8] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Hossein A. Rahmani, Daniel Campos, Jimmy Lin, Ellen M. Voorhees, and Ian Soboroff. 2024. Overview of the TREC 2023 Deep Learning Track. In *Text REtrieval Conference (TREC)*. NIST, TREC.

[9] W. Bruce Croft, Donald Metzler, and Trevor Strohman. 2009. *Search Engines: Information Retrieval in Practice*. Addison Wesley. https://www.amazon.com/Search-Engines-Information-Retrieval-Practice/dp/0136072240

[10] Laura Dietz. 2024. A Workbench for Autograding Retrieve/Generate Systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*. ACM, 1963–1972. https://doi.org/10.1145/3626772.3657871

[11] Laura Dietz. 2024. A workbench for autograding retrieve/generate systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1963–1972.

[12] Guglielmo Faggioli, Laura Dietz, Charles LA Clarke, Gianluca Demartini, Matthias Hagen, Claudia Hauff, Noriko Kando, Evangelos Kanoulas, Martin Potthast, Benno Stein, et al. 2023. Perspectives on large language models for relevance judgment. In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval*. 39–50.

[13] Naghmeh Farzi and Laura Dietz. 2024. Best in Tau@ LLMJudge: Criteria-Based Relevance Evaluation with Llama3. *arXiv preprint arXiv:2410.14044* (2024).

[14] Naghmeh Farzi and Laura Dietz. 2024. Pencils Down! Automatic Rubric-based Evaluation of Retrieve/Generate Systems. In *Proceedings of the 2024 ACM SIGIR International Conference on Theory of Information Retrieval* (Washington DC, USA) *(ICTIR '24)*. Association for Computing Machinery, New York, NY, USA, 175–184. https://doi.org/10.1145/3664190.3672511

[15] Donna K. Harman (Ed.). 1992. *Proceedings of The First Text REtrieval Conference, TREC 1992, Gaithersburg, Maryland, USA, November 4-6, 1992*. NIST Special Publication, Vol. 500-207. National Institute of Standards and Technology (NIST). http://trec.nist.gov/pubs/trec1/t1_proceedings.html

[16] 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, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*. ACM, 781–789. https://doi.org/10.1145/3018661.3018699

[17] Karen Sparck Jones. 1995. Reflections on TREC. *Inf. Process. Manag.* 31, 3 (1995), 291–314. https://doi.org/10.1016/0306-4573(94)00048-8

[18] Noriko Kando, Kazuko Kuriyama, Toshihiko Nozue, Koji Eguchi, Hiroyuki Kato, and Souichiro Hidaka. 1999. Overview of IR tasks at the first NTCIR workshop. In *Proceedings of the first NTCIR workshop on research in Japanese text retrieval and term recognition*. 11–44.

[19] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110* (2022).

[20] Sean MacAvaney and Luca Soldaini. 2023. One-Shot Labeling for Automatic Relevance Estimation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*. ACM, 2230–2235. https://doi.org/10.1145/3539618.3592032

[21] Chuan Meng, Negar Arabzadeh, Arian Askari, Mohammad Aliannejadi, and Maarten de Rijke. 2024. Query performance prediction using relevance judgments generated by large language models. *arXiv preprint arXiv:2404.01012* (2024).

[22] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016 (CEUR Workshop Proceedings, Vol. 1773)*. CEUR-WS.org. https://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf

[23] Andrew Parry, Maik Fröbe, Sean MacAvaney, Martin Potthast, and Matthias Hagen. 2024. Analyzing Adversarial Attacks on Sequence-to-Sequence Relevance Models. In *Advances in Information Retrieval - 46th European Conference on Information Retrieval, ECIR 2024, Glasgow, UK, March 24-28, 2024, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 14609)*. Springer, 286–302. https://doi.org/10.1007/978-3-031-56060-6_19

[24] Carol Peters (Ed.). 2001. *Cross-Language Information Retrieval and Evaluation, Workshop of Cross-Language Evaluation Forum, CLEF 2000, Lisbon, Portugal, September 21-22, 2000, Revised Papers*. Lecture Notes in Computer Science, Vol. 2069. Springer. https://doi.org/10.1007/3-540-44645-1

[25] Hossein A Rahmani, Nick Craswell, Emine Yilmaz, Bhaskar Mitra, and Daniel Campos. 2024. Synthetic Test Collections for Retrieval Evaluation. *arXiv preprint arXiv:2405.07767* (2024).

[26] Hossein A Rahmani, Clemencia Siro, Mohammad Aliannejadi, Nick Craswell, Charles LA Clarke, Guglielmo Faggioli, Bhaskar Mitra, Paul Thomas, and Emine Yilmaz. 2024. Report on the 1st workshop on large language model for evaluation in information retrieval (llm4eval 2024) at sigir 2024. *arXiv preprint arXiv:2408.05388* (2024).

[27] Hossein A. Rahmani, Clemencia Siro, Mohammad Aliannejadi, Nick Craswell, Charles L. A. Clarke, Guglielmo Faggioli, Bhaskar Mitra, Paul Thomas, and Emine Yilmaz. 2024. LLM4Eval: Large Language Model for Evaluation in IR. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Washington DC, USA) *(SIGIR '24)*. Association for Computing Machinery, New York, NY, USA, 3040–3043. https://doi.org/10.1145/3626772.3657992

[28] Hossein A Rahmani, Emine Yilmaz, Nick Craswell, and Bhaskar Mitra. 2024. JudgeBlender: Ensembling Judgments for Automatic Relevance Assessment. *arXiv preprint arXiv:2412.13268* (2024).

[29] Hossein A Rahmani, Emine Yilmaz, Nick Craswell, Bhaskar Mitra, Paul Thomas, Charles LA Clarke, Mohammad Aliannejadi, Clemencia Siro, and Guglielmo Faggioli. 2024. Llmjudge: Llms for relevance judgments. *arXiv preprint arXiv:2408.08896* (2024).

[30] Mark Sanderson. 2010. Test Collection Based Evaluation of Information Retrieval Systems. *Found. Trends Inf. Retr.* 4, 4 (2010), 247–375. https://doi.org/10.1561/1500000009

[31] Harrisen Scells, Shengyao Zhuang, and Guido Zuccon. 2022. Reduce, Reuse, Recycle: Green Information Retrieval Research. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*. ACM, 2825–2837. https://doi.org/10.1145/3477495.3531766

[32] Ian Soboroff. 2024. Don't Use LLMs to Make Relevance Judgments. *CoRR* abs/2409.15133 (2024). https://doi.org/10.48550/ARXIV.2409.15133 arXiv:2409.15133

[33] Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agent. *ArXiv* abs/2304.09542 (2023).

[34] Paul Thomas, Seth Spielman, Nick Craswell, and Bhaskar Mitra. 2023. Large language models can accurately predict searcher preferences. *arXiv preprint arXiv:2309.10621* (2023).

[35] Shivani Upadhyay, Ronak Pradeep, Nandan Thakur, Nick Craswell, and Jimmy Lin. 2024. UMBRELA: UMbrela is the (Open-Source Reproduction of the) Bing RELevance Assessor. *arXiv preprint arXiv:2406.06519* (2024).

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[37] Haijin Wang, Mianrong Zhang, Zheng Chen, Nan Shang, Shangheng Yao, Fushuan Wen, and Junhua Zhao. 2024. Carbon Footprint Accounting Driven by Large Language Models and Retrieval-augmented Generation. *CoRR* abs/2408.09713 (2024). https://doi.org/10.48550/ARXIV.2408.09713 arXiv:2408.09713

[38] Guido Zuccon, Harrisen Scells, and Shengyao Zhuang. 2023. Beyond CO2 Emissions: The Overlooked Impact of Water Consumption of Information Retrieval Models. In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR 2023, Taipei, Taiwan, 23 July 2023*. ACM, 283–289. https://doi.org/10.1145/3578337.3605121