LUPINE PUBLISHERS

Current Trends in Computer Sciences & Applications

DOI: 10.32474/CTCSA.2024.03.000160

ISSN: 2643-6744

Review Article

A Prompt-Based Evaluator for Llms: Making use of Chain of Thought Reasoning

Ayse Arslan*

Department of Computer Science, Oxford Alumni, Northern California

***Corresponding author:** Ayse Arslan, Department of Computer Science, Oxford Alumni of Northern California

Received: February 09, 2024 **Published:** February 27, 2024

Abstract

Large-scale transformers such as Chat GPT [57] and GPT4 [58] demonstrate unprecedented capabilities and impressive successes on seemingly complex tasks. Yet, they also display astonishing failures on seemingly trivial tasks. It is still not known under what conditions do transformers succeed, fail, and why. Seeking thorough answers to these questions remains an open research challenge. Therefore, this study explores a framework which is a prompt-based evaluator with three main components. It tries to overcome this gap between theory and practice by presenting an evaluation framework for LLMs. While the framework is theoretical in nature it offers a ground for future discussions about how to evaluate LLMs.

Introduction

Large Language Models (LLMs) are enabling more natural and sophisticated interactions between human-beings and machines, enhancing user experience in existing applications like coding [1], web search [2], chatbots [3,4], customer service and content creation. This transformation brought by LLMs is also paving the way for new innovative AI applications. While large-scale transformers such as Chat GPT [57] and GPT4 [58] demonstrate impressive successes on seemingly complex tasks they also display astonishing failures on seemingly trivial tasks which spark critical open questions about how to faithfully interpret their mixed capabilities. These problems present compelling challenges for AI systems as they require combining basic reasoning operations to follow computational paths that arrive at unique correct solutions. Under what conditions do transformers succeed, fail, and why? Can transformers be taught to follow reasoning paths? Seeking thorough answers to these questions remains an open research challenge. This study explores a framework which is a prompt-based evaluator with three main components. The prompt is a natural language instruction that defines the evaluation task and the desired evaluation criteria. Before going into technical details, the study provides an overview of LLMs. Next, it explores the use of the LLM evaluation framework.

Overview of LLMs

Knowledge is a fundamental component of human civilization. Throughout our lives, human-beings continuously gather an extensive wealth of knowledge and learn to adaptively apply it in various contexts. The enduring exploration of the nature of knowledge, and the processes by which we acquire, retain, and interpret it, continues to captivate scientists, which is not just a technical pursuit but a journey towards mirroring the nuanced complexities of human cognition, communication and intelligence [5]. Recently, Large Language Models (LLMs) like GPT-4 [6] have showcased a remarkable ability in Natural Language Processing (NLP) to retain a vast amount of knowledge, arguably surpassing human capacity. Large language models (LLMs) have demonstrated remarkable capabilities in instruction following and few-shot in-context learning (Brown et al., 2020). Large Language Models (LLMs) can not only summarize documents and converse on a large range of topics [5], but they have also shown other emergent abilities [1,7]. Traditionally, LLMs are provided with a context as a textual

prompt and are asked to provide answers via text completion, thereby solving a variety of choice-based [8], description-based [9], and reasoning tasks [6]. This achievement can be attributed to the way LLMs process and compress huge amount of data [1], potentially forming more concise, coherent, and interpretable models of the underlying generative processes, essentially creating a kind of "world model" [8].

An LLM has the following structure:

1. **Open-source LLM:** These are small open-source alternatives to Chat GPT which are trained on large amounts of text and can generate high-quality responses to user prompts.

2. **Embedding model:** An embedding model is used to transform text data into a numerical format that can be easily compared to other text data. This is typically done using a technique called word or sentence embeddings, which represent text as dense vectors in a high-dimensional space.

3. **Vector database:** A vector database is designed to store and retrieve embeddings. It can store the content of documents in a format that can be easily compared to the user's prompt.

4. **Knowledge documents:** This is a collection of documents that contain the knowledge an LLM will use to answer questions. It can be a collection of PDF or text documents that contain personal blog posts.

5. **User interface:** The user interface layer will take user prompts and display the model's output. This can be a simple command-line interface (CLI) or a more sophisticated web application. The user interface will send the user's prompt to the application and return he model's response to the user.

LLMs have limitations like factual fallacy, potential generation of harmful content, and outdated knowledge due to their training cut-off. A major limitation of LLMs is that they lack awareness of recent events and private knowledge. This issue can be partly mitigated by augmenting LLMs with information retrieved from external sources, a technique known as retrieval-augmented generation (RAG). On the other hand, LLMs can also serve as foundation models to enhance text embeddings. Retraining to correct these issues is both costly and time-consuming [8]. To address this, recent years have seen a surge in the development of knowledge editing techniques specifically tailored for LLMs, which allows for cost-effective post-hoc modifications to models [2]. This technique focuses on specific areas for adjustment without compromising overall performance and can help understand how LLMs represent and process information, which is crucial for ensuring the fairness, and safety in Artificial Intelligence (AI) applications [2].

Knowledge editing for LLMs can be classified into the following groups [9]:

- Resorting to External Knowledge. This kind of approach is similar to the recognition phase in human cognitive processes, which needs to be exposed to the new knowledge within a relevant context, just as people first encounter new information.

- Merging Knowledge into the Model. This kind of approach closely resembles the association phrase in human cognitive processes, in which connections are formed between the new knowledge and existing knowledge in the model. Methods would combine or substitute the output or intermediate output with a learned knowledge representation.

- Editing Intrinsic Knowledge. This approach to knowledge editing is akin to the mastery phase in human cognitive processes. It involves the model fully integrating knowledge into its parameters by modifying the weights and utilizing them reliably.

The recent NLP literature has witnessed a tremendous amount of activity in building models that can follow natural language.

To begin with, instruction tuning involves learning from input-output pairs where the input is natural language task description, and the output is a demonstration of the desired behavior. Instruction tuning has been shown to improve the model's ability to follow instructions on both seen and unseen tasks [10], improve the overall quality of the generations [10] and give models enhanced zero-shot and reasoning abilities [2].

A general-purpose instruction-following model involves one or more of these three techniques:

1. Plain language model prompting, where one prepares an incomplete text such that a typical continuation of the text should represent a completion of the intended task.

2. Supervised fine-tuning, where one trains the model to match high-quality human demonstrations on the task.

3. Reinforcement learning, where one incrementally weakens or strengthens certain model behaviors according to preference judgments from a human tester or user.

However, these techniques cannot guarantee that an AI model will behave appropriately in every plausible situation it will face in deployment. Nor can they even make a model try to behave appropriately to the extent possible given its skills and knowledge (to the extent that it can be said to have generalizable skills or knowledge) Figure 1. As LLMs are trained on large corpuses of data these models may increase the risk of misinformation, privacy violations, socioeconomic harms, and representational harms. Prompting can help mitigate these concerns by guiding AI-generated content towards more accurate, ethical, and contextually appropriate outputs. At a very high level, the process of prompting can be described as follows:

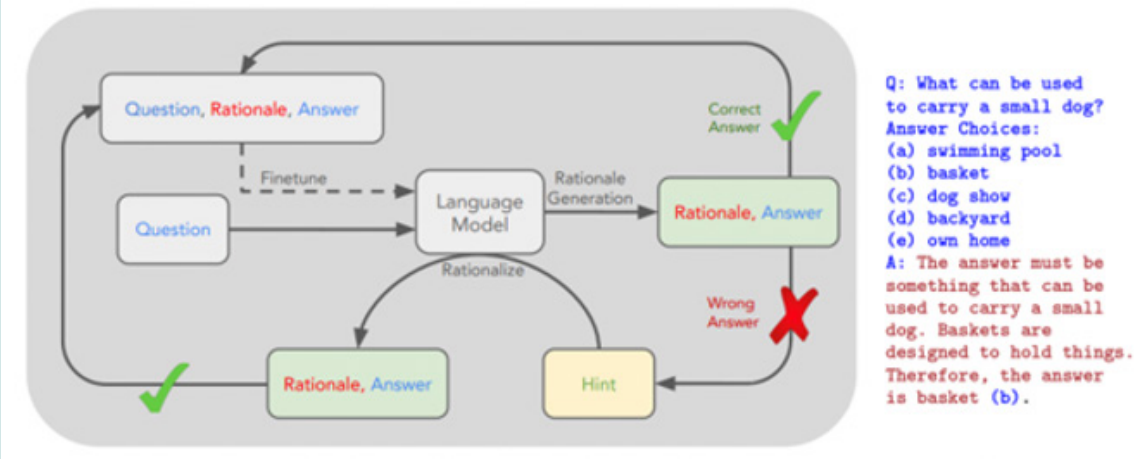

Figure 1: Overview of Selection-Inference (Exploiting Large Language Models for Interpretable Logical Reasoning by Antonia Creswell et al. (2022)).

1. The user enters a prompt in the user interface.

2. The application uses the embedding model to create an embedding from the user's prompt and send it to the vector database.

3. The vector database returns a list of documents that are relevant to the prompt based on the similarity of their embeddings to the user's prompt.

4. The application creates a new prompt with the user's initial prompt and the retrieved documents as context and sends it to the local LLM.

5. The LLM produces the result along with citations from the context documents. The result is displayed in the user interface along with the sources.

Observing that an LLM performs a task successfully in one instance is not strong evidence that the LLM is capable of performing that task in general, especially if that example was cherry-picked as part of a demonstration. On the other hand, there is also increasingly substantial evidence that LLMs develop internal representations of the world to some extent:

- Models can make inferences about what the author of a document knows or believes and use these inferences to predict how the document will be continued.

- Models use internal representations of the properties and locations of objects described in stories, which evolve as more information about these objects is revealed.

- Models can distinguish common misconceptions from true facts, and often show well calibrated internal representations for how likely a claim is to be true.

- Models pass many tests designed to measure commonsense reasoning. These results are in tension, at least to some extent, with the common intuition that LLMs are nothing but statistical next-word predictors, and therefore cannot learn or reason about anything but text.

Moreover, techniques such as the chain-of-thought reasoning strategies can further aid a model to improve its internal representation. Simply prompting a model to "think step by step" can lead it to perform well on entire categories of math and reasoning problems that it would otherwise fail on.

Published by Antonia Creswell et al., one extension of the chain-of-thought technique is to split the single prompt for generating explanations and answers into smaller parts. First, a prompt selects a relevant subset of facts from the text ('Selection prompt'). Then, a second prompt infers a conclusion from the selected facts ('Inference prompt'). Moreover, propose Chain of Thought (CoT) Prompting, a technique that triggers the model to generate a rationale before the answer. By generating a rationale, large LMs show improved reasoning abilities when solving challenging tasks. show that by appending the phrase 'Let's think step by step', large LMs could perform CoT prompting in a zero-shot setting.

As shown in Figure 2, by only feeding the Task Introduction and the Evaluation Criteria as a prompt, one asks LLMs to generate a CoT of detailed Evaluation Steps. Then, one uses the prompt along with the generated CoT to evaluate the outputs. The evaluator output is formatted as a form. Moreover, the probabilities of the output rating tokens can be used to refine the final metric having a bias towards the LLM-generated texts.

The framework is a prompt-based evaluator with three main components:

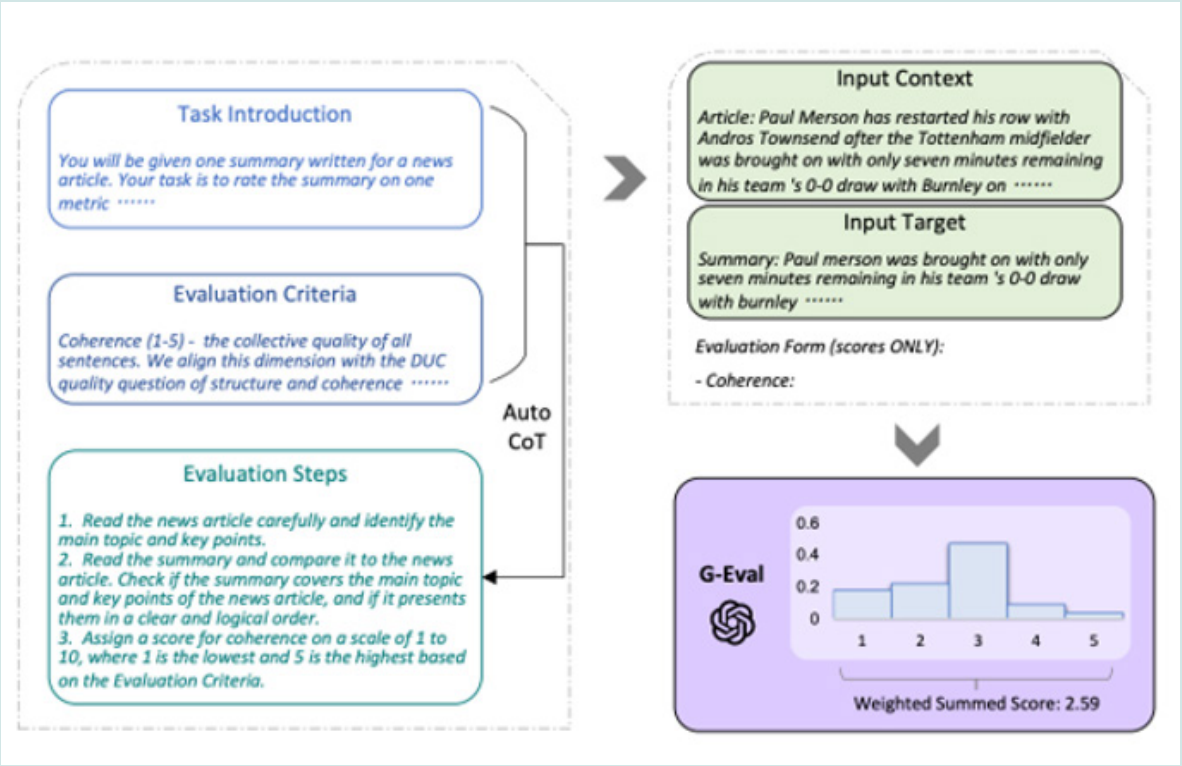

Figure 2: The Overall Framework for LLM Evaluation.

1) a prompt that contains the definition of the evaluation task and the desired evaluation criteria,

2) a chain-of-thoughts (CoT) that is a set of intermediate instructions generated by the LLM describing the detailed evaluation steps, and

3) a scoring function that calls LLM and calculates the score based on the probabilities of the return tokens.

The prompt is a natural language instruction that defines the evaluation task and the desired evaluation criteria. For example, for text summarization, the prompt can be: "You will be given one summary written for a news article. Your task is to rate the summary on one metric. Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed." The chain-of-thoughts (CoT) is a sequence of intermediate representations that are generated by the LLM during the text generation process. For evaluation tasks, some criteria need a more detailed evaluation instruction beyond the simple definition, and it is time-consuming to manually design such evaluation steps for each task. The CoT can provide more context and guidance for the LLM to evaluate the generated text, and can also help to explain the evaluation process and results.

For example, for evaluating coherence in text summarization, one can add a line of "Evaluation Steps:" to the prompt and let LLM to generate the following CoT automatically:

1. Read the news article carefully and identify the main topic and key points.

2. Read the summary and compare it to the news article. Check if the summary covers the main topic and key points of the news article, and if it presents them in a clear and logical order.

3. Assign a score for coherence on a scale of 1 to 5, where 1 is the lowest and 5 is the highest based on the Evaluation Criteria.

The scoring function calls the LLM with the designed prompt, auto CoT, the input context and the target text that needs to be evaluated. Unlike GPT Score which uses the conditional probability of generating the target text as an evaluation metric, G-EVAL directly performs the evaluation task with a form-filling paradigm. While CoT prompting works effectively for large LMs with more than 100 billion parameters, it does not necessarily confer the same benefits to smaller LMs. The requirement of a large number of parameters consequently results in significant computational cost and accessibility issues. Increasingly capable LLMs, with increasingly accurate and usable internal models of the world, are likely to be able to take on increasingly open-ended tasks that involve making and executing novel plans to optimize for outcomes in the world. It should also be noted that while researchers can evaluate whether systems are effective or ineffective, reliable or unreliable, interpretable or uninterpretable, questions of consciousness, sentience, rights, and moral patient hood in LLMs, are worth distinguishing from the issues above.

Evaluating AI

Evaluation, in the context of AI, involves measuring system performance or impact, with results compared against a normative baseline, determining whether the AI system is deemed "good," "fair," or "safe enough." However, a sociotechnical gap arises when safety evaluations focus solely on the technical aspects, neglecting human and systemic factors. Socio-technical research plays a crucial role in broadening the scope of AI system evaluation, incorporating human and systemic elements. Recognizing AI systems as socio-technical entities, this approach emphasizes the inherent value systems embedded in design choices, highlighting the need for effective governance and recourse mechanisms.

Designing an evaluation involves explicit or implicit choices on what to prioritize and discard. Selecting a target for evaluation necessitates a normative judgment on what harms are significant. The overall process can be structured as follows:

- Grounding Operationalization: An evaluation can be grounded in a literature review, human annotation, or expert-curated examples, with diverse perspectives for testing operationalizations.

- Documenting Limitations: As risks of harm are latent concepts, documenting and signposting limitations allow others to interpret results better, acknowledging the choices made during operationalization.

- Cross-Validating Operationalization: Cross-validating by comparing results from different evaluations of the same concept identifies areas where metrics operationalize harm differently.

- Existing Taxonomies and Research: Existing taxonomies address risks from AI systems in audio and text. Overview harms from generative AI systems and describe social impact analysis approaches for identified harm areas. Research highlights the capability of large language models in generating factual information, termed 'factuality.' Chain-of-thought generation, especially evident in models like Chat GPT, enhances accuracy in complex reasoning tasks, such as solving math problems.

As AI models like LLMs become integral to services like search engines, chatbots, and content generators, ensuring factual accuracy is crucial to prevent misinformation and potential harm.

Three-Layered Framework for Safety Evaluations:

Google's recent study presents a three-layered framework for safety evaluations of AI systems: capability evaluation, human interaction evaluation, and systemic impact evaluation. These layers progressively add contextual layers critical for assessing risks of harm.

Inspecting the state of evaluations applied to generative AI systems reveals three high-level gaps:

- Coverage Gap: Evaluations for several risks are lacking, especially in social risk evaluation. Gaps exist where few or no evaluations assess a specific risk area.

- Context Gap: Human interaction and systemic evaluations are rare, with existing evaluations predominantly focused on the text modality, leaving gaps in audio, image, video, or combined modalities.

- Multimodal Gap: Evaluations are missing for multimodal AI systems, with most evaluations concentrating on capability evaluations.

The three layers in Google's framework interact, with their boundaries being gradual. Observations at one layer may indicate related observations at the next, emphasizing the importance of a comprehensive evaluation approach

As shown in Figure 2, the framework consists of the following layers:

1. Layer 1: Capability

Capabilities include metrics that are designed to track efficiency and can be assessed against fixed, automated tests or probed dynamically by human or automated adversarial testers.

Evaluations at this layer can also concern the data on which a model is trained. Capability evaluation is critical, but insufficient, for a comprehensive safety evaluation. It can serve as an early indicator of potential downstream harms, but to assess whether or not a capability relates to risks of harm requires taking into account context – such as who uses the AI system, to what end, and under which circumstances. This context is assessed at subsequent layers.

2. Layer 2: Human Interaction

This layer centers the experience of people interacting with a given AI system. It also includes evaluating processes by which these artefacts are created, such as the aggregation mechanisms in processes that are used to adapt an AI system to a particular task. Several risks of harm can be evaluated by measuring capabilities through the outputs of an AI system. This includes, for example, the extent to which an AI model reproduces harmful stereotypes in images or utterances (representation harms), or makes factual errors. This can be done by considering the following questions: Does the AI system perform its intended function at the point of use?

- How do experiences differ between user groups?

- Does human–AI interaction lead to unintended effects on the person interacting or exposed to AI outputs? Evaluation that considers an AI system in the context of use can assess the overall performance of the human–AI dyad, such as quality of outcomes on AI-assisted computer coding tasks compared to a human–human. While this layer provides critical context by adding human interaction to the evaluation, it remains insufficient for a comprehensive AI safety assessment. Assessing these effects requires analyzing the broader systems

into which an AI system is deployed, at the third and final layer of our sociotechnical framework for safety evaluation.

3. Layer 3: Systemic impact

Widely used AI systems shape, and are shaped by, the societies in which they are used.

Impact from generative AI systems on societal institutions, such as political polarization or changes to trust in public media, can be evaluated through system evaluation.

Systemic impacts are often difficult to assess due to the complex nature, idiosyncrasies, and noise of the systems that are being evaluated. While direct impacts of an AI system may not be known until post deployment, forecasts or comparable technologies can provide initial insights on potential risks of harm at this layer.

Limitations

Modern large language models (LLMs) models are of course not problem-free. Among their unfavorable behaviors it is possible to find toxicity, bias, and hallucination. One of the settings in which LLMs are notoriously prone to hallucinate is when presented with (un)answerable questions. Recent works in this setting, suggested using models' confidence as an indication of answerability, and some suggested further finetuning to enhance the probability of detecting (un)answerable questions. (Un-)answerability capabilities in LLMs were mainly studied by using few-shot prompting. Moreover, several works have recently shown that LLMs become easier to steer with natural language prompts either as they become larger or as they are exposed to larger instruction tuning data, and as a consequence, it might improve the (un)answerability capabilities of the model. Automatic prompt-tuning can be also used for improving (un)answerability capabilities, without the need for manual handcrafting prompts. Introduced a prompt tuning-based strategy to mitigate (un)answerable questions, by mapping questions into their proper, specific templates Figure 3.

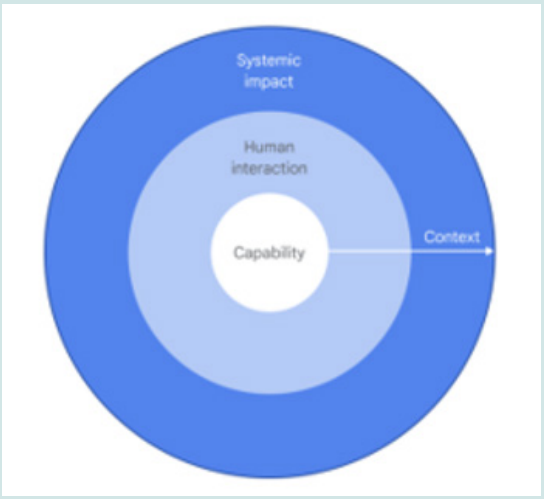

Figure 3: Three-Layer Framework (Google, 2023).

Conclusion

In general, the AI community still lacks a comprehensive strategy to fully leverage CoT prompting to solve multiple unseen novel tasks in the context of smaller LMs. Recent work has focused on empowering relatively smaller language models to effectively solve novel tasks as well, primarily through fine-tuning with rationales (denoted as CoT fine-tuning) and applying CoT prompting on a single target task. However, solving a single task does not adequately address the issue of generalization to a broad range of unseen tasks. This study tries to overcome this gap between theory and practice by presenting an evaluation framework for LLMs. While the framework is theoretical in nature it offers a ground for future discussions about how to evaluate LLMs.

References

1. Gulli A (2013) A deeper look at Autosuggest, Microsoft Bing Blogs.
2. Olteanu, C Castillo, J Boy, K Varshey (2018) The effect of extremist violence on hateful speech online, || Proceedings of the Twelfth International AAAI Conference on Web and social media 12(1): 1-11.
3. Mc Guffie and A Newhouse (2020) The radicalization risks of GPT-3 and advanced neural language models pp.1-13.
4. Miller and I Record M (2017) Responsible epistemic technologies: A social-epistemological analysis of autocompleted Web search, New Media & Society 19(12): 1945-1963.
5. Akhtar (2016) Google defends its search engine against charges it favors Clinton.
6. H Yenala, M Chinnakotla, J Goyal (2017) Convolutional bi-directional LSTM for detecting inappropriate query suggestions in Web search,|| In: J Kim, K Shim, L Cao, JG Lee, X Lin, YS Moon (editors). Advances in knowledge discovery and data mining. Lecture Notes in Computer Science, volume 10234. Cham, Switzerland: Springer, pp. 3-16.
7. Arentz W and B Olstad (2016) Classifying offensive sites based on image content,|| Computer Vision and Image Understanding 94(1-3): 295-310.

8. Y Shen, X He, J Gao, L Deng, G Mesnil (2017) Learning semantic representations using convolutional neural networks for Web search,|| WWW '14 Companion: Proceedings of the 23rd International Conference on World Wide Web pp. 373-374.

9. Olteanu, C Castillo, F Diaz, E Kcman (2019) Social data: Biases, methodological pitfalls, and ethical boundaries,|| Frontiers in Big Data 2(1).

10. Olteanu, K Talamadupula, K Varshney (2017) The limits of abstract evaluation metrics: The case of hate speech detection, || WebSci '17: Proceedings of the 2017 ACM on Web Science Conference pp. 405-406.

DOI: 10.32474/CTCSA.2024.03.000160